NASA Contractor Report 3662

The NYU Inverse Swept Wing Code

F. Bauer, P. Garabedian,
and G. McFadden

GRANT NSG-1579
JANUARY 1983

NASA

NASA Contractor Report 3662

The NYU Inverse Swept Wing Code

F. Bauer, P. Garabedian,
and G. McFadden
Courant Institute of Mathematical Sciences
New York University, New York, New York

Prepared for
Langley Research Center
under Grant NSG-1579

NASA
National Aeronautics
and Space Administration

**Scientific and Technical
Information Branch**

1983

TABLE OF CONTENTS

NOMENCLATURE

a_1, a_2, a_3, a_4	Coefficients of free boundary equation
a, b	Limits of integration
A	Aspect ratio
B	Factor in drag formula
c	Speed of sound
C	Slope
C_D	Drag coefficient
C_{DW}	Wave drag coefficient
C_L	Lift coefficient
C_P	Pressure coefficient
f	Free surface of wing
f_0	Fixed underlying surface
G	Reduced potential
h	Mesh size
i,j,k	Indices
M	Mach number
q	Speed
q_0	Prescribed speed
Q	Free boundary formula
r	Radius
s	Coordinate in the direction of flow
u,v,w	Velocity components
x,y,z	Rectangular coordinates
X,Y,Z	Mapped coordinates
α_1, α_2	Assigned values of ϕ
β	Scale factor in boundary condition
γ	Gas constant
ϕ	Velocity potential
ψ	Stream function

SUMMARY

An inverse swept wing code is described that is based on the widely used transonic flow program FLO22. The new code incorporates a free boundary algorithm permitting the pressure distribution to be prescribed over a portion of the wing surface. A special routine is included to calculate the wave drag, which can be minimized in its dependence on the pressure distribution. An alternate formulation of the boundary condition at infinity has been introduced to enhance the speed and accuracy of the code. A FORTRAN listing of the code and a listing of a sample run are presented. There is also a user's manual as well as glossaries of input and output parameters.

INTRODUCTION

After much controversy about shockless airfoils in the theory of transonic flow, experimental work has established that wings can be constructed to virtually eliminate wave drag over a practical range of supercritical speeds. Computational fluid dynamics has become a primary tool for the design and analysis of these supercritical wings. More specifically, computer codes developed at NYU to calculate transonic flow in both two and three dimensions have become widely accepted by the aircraft industry. It is our purpose here to describe and list the latest of these codes, which serves to redesign a swept wing by selecting its pressure distribution so that the wave drag is minimized at a fixed speed and angle of attack.

Perhaps the best way to design shockless airfoils in two-dimensional transonic flow is to use the hodograph transformation in combination with analytic continuation into the complex domain [1,3]. Most analysis codes, on the other hand, depend on an introduction of artificial viscosity and artificial time that is motivated by the retarded difference scheme of Murman and Cole [9]. The method of artificial viscosity can also be applied to the design problem, for which it is especially helpful in three dimensions [6,7]. An approach of this kind has been adopted in developing the inverse swept wing code we are now concerned with. Our procedure has been to modify the FLO22 code of Jameson and Caughey, which is in turn based on an earlier oblique wing code for the calculation of transonic flow in three dimensions [2,8].

In the next section of the report we shall review theoretical aspects of the transonics codes that are either somewhat controversial or have not been well publicized elsewhere.

Indications will be given of how the basic method can be generalized; but detailed treatments of more complicated problems, such as the three-dimensional flow through a cascade of compressor blades, will be left to other publications. An example of a supercritical swept wing that has been redesigned by applying the new three-dimensional code will be discussed. Both computational and physical properties of the example will be emphasized. Then a detailed description of the code will be presented that can serve as a user's manual. The final sections of the report are devoted to the listing of a sample run for the supercritical wing just referred to and to a listing of the code, with comment cards.

MATHEMATICAL BACKGROUND

The transonic flow around airfoils and wings is usually calculated by considering a velocity potential ϕ that satisfies the second order quasilinear partial differential equation

$$(c^2-u^2)\phi_{xx} + (c^2-v^2)\phi_{yy} + (c^2-w^2)\phi_{zz} - 2uv\phi_{xy} - 2vw\phi_{yz} - 2wu\phi_{zx} = 0 \quad ,$$

where $u = \phi_z$, $v = \phi_y$ and $w = \phi_z$ are the velocity components and c is the speed of sound defined by Bernoulli's law

$$\frac{u^2 + v^2 + w^2}{2} + \frac{c^2}{\gamma - 1} = \text{const.}$$

A Neumann problem for ϕ is specified by setting its normal derivative equal to zero at the wing and prescribing its asymptotic behavior at infinity. Finite difference schemes that capture weak shock waves effectively are arrived at by adding an artificial viscosity term to the equation for ϕ. This term is obtained by retarding difference approximations to second derivatives in the direction of the flow, which does not alter the boundary condition at the wing. Iterative methods to solve the difference equations for ϕ are found by introducing artificially time-dependent terms that force decay to a steady state [2].

An objection can be made to use of the velocity potential because that presumes constant entropy, whereas the wave drag, which is of primary interest, has the same order of magnitude as the jump in entropy across shocks, to which it can even be attributed. However, we have been able to develop an expression for the wave drag in terms of the velocity potential that is accurate to lowest order for weak normal shock waves [6]. This is important because there are ambiguities in determining a steady state solution of the Euler equations that are perhaps

best overcome by the assumption of irrotationality that characterizes potential flow. More general steady solutions may include vortices such as those that occur in models of the wake. Therefore some hypothesis must be made to ensure uniqueness of the flow.

In two dimensions another possibility for handling the Euler equations is to introduce the stream function ψ as an independent variable and to calculate the flow by solving a partial differential equation for the ordinate y as a function

$$y = y(x,\psi)$$

of x and ψ. This is equivalent to making the topological assumption that each vertical line intersects each streamline just once, which does eliminate vortices. But it is awkward to formulate the laws of conservation of momentum in a fashion convenient for numerical computation of the unknown y. Furthermore, experience with analogous problems in the calculations of magnetohydrodynamic equilibrium shows that existence as well as uniqueness becomes questionable for steady solutions of the Euler equations in three dimensions. The analogy is based on letting the velocity, the vorticity and the Bernoulli constant for incompressible flow correspond respectively to the magnetic field, the current density and the pressure in magnetohydrodynamics [5].

How the wave drag may be represented in terms of the velocity potential ϕ is most readily understood by studying a model problem for one-dimensional flow. Application of the retarded difference scheme of Murman and Cole to the small disturbance equation for ϕ leads us to consider the ordinary differential equation

$$\frac{1}{2}\,[\phi_x^2]_x = h[\max(\phi_x,0)\phi_{xx}]_x$$

describing conservation of mass, where h is a positive mesh size parameter multiplying a term on the right that we conceive of as artificial viscosity. The flow is said to be supersonic when $\phi_x > 0$ and subsonic when $\phi_x < 0$. If appropriate boundary conditions of the form

$$\phi(a) = \alpha_1 , \qquad \phi(b) = \alpha_2 , \qquad \phi_x(a) = C > 0$$

are imposed at the ends of the interval [a,b], a unique solution is found that approaches a pair of intersecting lines with the opposite slopes C and -C as $h \to 0$. The intersection of the lines is a shock wave across which ϕ_x^2 remains continuous [2].

Multiplying by ϕ_x on both sides of the ordinary differential equation for ϕ, we obtain an analogue

$$\frac{1}{3}\,[\phi_x^3]_x = h[\max(\phi_x,0)\phi_x\phi_{xx}]_x - h\max(\phi_x,0)\phi_{xx}^2$$

of the law of conservation of momentum. Integration by parts and passage to the limit as $h \to 0$ yields the entropy inequality

$$-\frac{2}{3}C^3 = \frac{1}{3}\phi_x(b)^3 - \frac{1}{3}\phi_x(a)^3$$
$$= -\lim_{h\to 0} h\int_a^b \max(\phi_x,0)\phi_{xx}^2\,dx \le 0\ .$$

This not only establishes the necessity of the requirement $C > 0$ in our model problem, but also suggests that the integral on the right is a legitimate measure of both the wave drag and the jump in entropy. A similar argument has been used to represent the wave drag as a volume integral involving ϕ for potential flow in both two and three dimensions [6,7]. The resulting formula has been implemented in our swept wing code and enables us to plot shock waves in a fashion indicating the amount of drag associated with them.

It is important to realize that the integrand in the volume integral for the wave drag depends in a subtle way on the form of the artificial viscosity used to calculate ϕ. To understand why this should be so one has only to alter the artificial viscosity on the right in the ordinary differential equation given above for ϕ to obtain

$$\phi_x\phi_{xx} = h\,\phi_{xxx}$$

instead. The same solution as before is found in the limit as $h \to 0$. The resulting entropy inequality

$$-\frac{2}{3}C^3 = -\lim_{h\to 0} h\int_a^b \phi_{xx}^2\,dx \le 0$$

remains unaltered except that there is a change in the integrand on the right. Thus the integral representing the wave drag is seen to have the same value it had before, but the way in which the shock wave is smeared when $h > 0$ becomes significantly different.

Another issue that arises in the computation of transonic flow around airfoils and wings is whether or not to put the finite difference equations in conservation form. Strictly

speaking this must be done to approximate the shock conditions accurately. However, boundary layer-shock wave interaction and, more specifically, the pressure recovery at the foot of a normal shock wave are poorly modeled by the conservation form of the equation for ϕ. This seems to be due to a term in the artificial viscosity that can be eliminated by reverting to a simpler difference scheme that is closely related to the original method of Murman and Cole. We have chosen to retain such a nonconservative scheme in the swept wing code listed in this report. However, it is not difficult to modify the code so as to bring the equation for ϕ into the mathematically more correct conservation form.

For the model problem of one-dimensional flow the artificial viscosity in conservation form is

$$h[\phi_x \phi_{xx}]_x = h\ \phi_x \phi_{xxx} + h\ \phi_{xx}^2 ,$$

whereas the nonconservative version is just $h\ \phi_x \phi_{xxx}$. The difference between these two viscosities is a positive term $h\phi_{xx}^2$ referred to above that represents mass generated by shock waves. For the full transonic flow problem the analogous quantity may contribute significantly to truncation error in supersonic regions where no shocks occur. Its omission from the nonconservative scheme adopted in the swept wing code therefore has the advantage of improving accuracy to a certain extent on the crude meshes that one must resort to for a three-dimensional calculation of this kind. Moreover, the nonconservative scheme seems especially appropriate for flows that are designed to be shockless anyway.

Our principal concern is the inverse problem of shaping a swept wing so that its pressure distribution may be prescribed. More specifically, we wish to choose the surface $y = f(x,z)$ of the wing so that the square of the speed q assumes given values $q_0(x,z)^2$. This requirement yields a free boundary condition

$$Q(f,f_x,f_z) = q_0(x,z)^2 - q^2 = 0$$

which we may view as a partial differential equation of the first order for the unknown function f. In the implementation of the computer code x, y and z are taken to be sheared parabolic coordinates such that the surface of the wing lies near the plane $y = 0$ and the flow is restricted to the half-plane $y > 0$. Problems with closure are circumvented by introducing a fixed surface $y = f_0(x,z)$ and imposing the constraint $f \geq f_0$, which

asserts that the wing must enclose a specified inner structure. The free boundary condition is only supposed to be fulfilled at points where $f > f_0$. Difficulties in locating stagnation at the leading edge or with closure at the trailing edge are avoided by choosing the assigned speed q_0 so that it decays rapidly there and makes $f = f_0$ outside a range in the middle of the wing where the free boundary condition becomes operative.

The free boundary problem we have formulated seems to be well posed even in the case of transonic flow, but hanging shocks tend to appear above the wing even when the prescribed pressure distribution is smooth at the surface. An iterative scheme to solve the free boundary problem numerically is arrived at by letting the free surface function f vary suitably with the artificial time parameter t of the transonic flow calculation. The motion of the free surface is defined by requiring that a partial differential equation of the form

$$a_1 f_{xt} + a_2 f_t = Q + a_3 Q_{f_x} Q_x + a_4 Q_{f_z} Q_z$$

be satisfied at points where $f > f_0$. The coefficients a_j are adjusted to achieve convergence of the method. An analogue of the Lax-Wendroff finite difference scheme is used for the computation of f. Precisely how this is accomplished is a more subtle matter concerning which the reader is referred to the listing of the code.

To eliminate shock waves on a swept wing at given speed and angle of attack it does not suffice just to prescribe the pressure smoothly. Experience with shockless airfoils designed by the hodograph method suggests that to suppress shocks the pressure distribution ought to be peaky at the front of the supersonic zone. We have incorporated in the code an exponential spline routine that generates desirable distributions of speed depending on relatively few free parameters [7]. There is also an option that enables one to choose the prescribed distribution so as to minimize the wave drag. The code can be used to design swept wings that achieve virtually shockless transonic flow at a specified condition. While this requires some skill because the problem of design is far from easy in practice, the code does seem to be robust and is capable of delivering meaningful results for a relatively modest expenditure of computational resources.

The speed and performance of the code have been improved by vectorization and by modifying the boundary condition at infinity. The new boundary condition is imposed on a control surface at some distance from the wing. It asserts that the reduced potential should decay at a specified rate as its argument approaches infinity. This is equivalent to a linear

combination of homogeneous Dirichlet and Neumann conditions on the control surface. The precise rate of decay has been adjusted semi-empirically to give optimal results.

The computational methods used to develop the inverse swept wing code can be applied to a variety of harder problems. Of particular interest are the flow past propellers or through cascades of compressor blades and the flow around airplane configurations that include a fuselage or engine nacelles. The most crucial issue is how to treat complicated geometries in space of three dimensions. While the question of adequate coordinate generation remains a challenge, it would appear that the difficulties associated with transonic flow are now well in hand. It may also be worth inquiring whether less directly related problems, such as that of ship wave resistance, might be attacked successfully by similar techniques of computational fluid dynamics.

DESIGN OF A STANDARD SUPERCRITICAL WING

Codes for the analysis of transonic flow around given bodies have been used quite successfully to simulate wind tunnel measurements, especially in the case of two-dimensional motion. The agreement between computed and experimental data is usually excellent, and the calculated results are obtained more quickly and sometimes more cheaply. In Fig. 1 we display a typical comparison of theoretical estimates of the drag coefficient C_D with experimental measurements that shows how well the new formula for the wave drag we have described works in practice for two-dimensional flow.

Design codes have been received less enthusiastically by the engineering community. They leave more choices up to the user, and the outcome of the computations may be less tangible. In this section we present an example of a supercritical wing that has been redesigned through an application of the inverse swept wing code. The results serve primarily as a sample run to illustrate how the code works, but they are also not without physical interest despite the crudeness of the mesh that is used.

It is best to construct the swept wing from a supercritical airfoil to begin with. For this purpose we have chosen a 13% thick airfoil that is shockless at free stream Mach number $M = 0.75$ and lift coefficient $C_L = 0.54$. The wing is swept back through an angle of 30° and has aspect ratio $A = 3.8$.

To redesign the wing, which has noticeable shocks near the wall, a typically shockless pressure distribution has been prescribed. It is specified in vertical sections by exponential

splines defined over three adjacent intervals. Optimization was used to select peak values of the pressure so that the wave drag became a minimum at the design condition of free stream Mach number M = 0.83 and lift coefficient $C_L = 0.41$. This reduced the wave drag coefficient C_{DW} from 0.0028 to 0.0008 and softened the shock waves perceptibly.

Fig. 2 shows how the plot routine of the code displays the results of an analysis calculation for our wing, and Fig. 3 shows corresponding data after the design run. Five sections of the wing are seen on the right, and corresponding distributions of the pressure coefficient C_P over the upper surface are seen on the left. Shock waves are plotted above the wing with a thickness indicative of the wave drag associated with them. The detailed input and output of the design run are listed in the report. The mesh consisted of $128 \times 10 \times 12$ points, and 100 iterations were performed to achieve acceptable convergence. This took 10 minutes of machine time on the CDC 6600 computer.

DESCRIPTION OF THE CODE

The NYU inverse swept wing code can be used for both analysis and design of a swept wing. In the design mode an option is available which invokes an optimizer (Harwell Mathematical Subroutine Library, AERE, England) to minimize the wave drag.

The analysis mode is like FLO22, which calculates the transonic flow past a given swept wing [8]. For analysis, data for the code is input on cards and stored on Tape 5 or read directly into Tape 5. The input consists of computation parameters, wing geometry, and physical specification of the flow. The resulting information is stored on Tape 7. This file can be saved and is used to initialize the computation if a wing is to be designed.

In the analysis mode the principal difference between the NYU inverse swept wing code and FLO22 is the introduction of a new boundary condition at infinity for the reduced potential G. This condition is imposed on an outer control surface and it improves the speed and accuracy of the computation. It has the form

$$G_{j+1} = (1 - \beta) G_j ,$$

where the index j+1 refers to a ghost point just outside the computational domain. The positive parameter β is scaled so that the requirement models a mixed Dirichlet and Neumann

condition

$$\frac{\partial G}{\partial r} + \frac{1}{r} G = 0$$

that serves to annihilate the leading term 1/r of a hypothetical expansion of G in spherical harmonics. In practice β has been chosen semi-empirically to give optimal resolution. The new boundary condition provides a more effective way of asserting that G decays at infinity [4,7].

In the design mode the surface of a given wing is modified so that a prescribed Mach number distribution is assumed over a portion of the wing. The optimization package attempts to minimize the wave drag by changing the Mach number distribution systematically. The wave drag is computed using a formula that has been discussed in the section on mathematical background.

More precisely, the factor ϕ_x occurring there is replaced by a term $M^2 - 1$ involving the Mach number M, and the derivative ϕ_{xx} is replaced by a second derivative ϕ_{ss} in the direction of the flow. This results in an expression of the form

$$C_{DW} = \sum Ah^4 \max(M^2-1,0)\ \phi_{ss}^2$$

for the wave drag coefficient C_{DW}, where the summation is extended over all mesh points and A is a factor determined by the finite difference equations that are used. Contributions from rarefaction waves are automatically excluded by the code.

To modify a given wing in the design mode, an analysis run for the unmodified, or baseline, wing is made to assess the characteristics of the flow field. The resulting speed distribution is examined and used to construct a more desirable distribution. The new distribution is input to the code on Tape 9. An exponential spline routine in the code calculates the desired distribution based on the input. This should have approximately the same spanwise distribution of lift as the original wing. It should be smooth throughout the supersonic zone, but may be peaky near the leading edge. In addition to the design distribution, a wing surface is prescribed that is identical to the original baseline wing near the leading and trailing edges but may be slightly thinner near the middle of each spanwise chord. The design scheme adds material to this underlying shape to fill in the thinned areas in a manner consistent with the assigned speed distribution. The thinning is done by introducing a groove. The parameters defining the shape of the groove are input to the code on Tape 9. To avoid difficulties with trailing edge

closure and maintenance of thickness-to-chord ratio, the surface modifications are made on a region of the upper surface that excludes the leading and trailing edges.

The computational domain has been obtained by applying a square root transformation to the physical domain that results in a representation Y = SO(X,Z) of the wing as a shallow bump lying above the (X,Z)-plane. The wing surface is changed iteratively, starting from the original shape as an initial guess. At each step one or more cycles of line relaxation are done in the standard analysis mode. The resulting speed distribution is compared with the desired distribution. Surface modifications are made that depend on the difference between the two distributions. If the modifications cause the computed surface to fall below the prescribed underlying surface Y = SOPRE(X,Z), then the computed surface is replaced by SOPRE(X,Z) at such points. The procedure is repeated with more line relaxations until the computed and assigned distributions agree. The surface modifications are determined from a first order partial differential equation that has been discussed earlier. Parameters controlling the scheme are discussed in the glossary. Assigning unrealistically low velocities near the leading and trailing edges serves to drive the computed surface onto the prescribed surface, which provides trailing edge closure and leaves the nose unaffected.

The design distribution is defined by two or more section Mach number distributions. Linear interpolation is used to specify the values elsewhere. The section speed distributions are assigned over the computational domain. For a fixed cross section Z the lower trailing edge appears on the extreme left, the leading edge appears near the center and has the largest values of SO, and the upper trailing edge appears on the extreme right.

The section distributions are defined by specifying input speeds Q1, Q2, Q3 and Q4 at fractions PCQ1, PCQ2, PCQ3 and PCQ4 = 1 of the local chord and by interpolating in between with an exponential spline. The spline has free parameters that can be adjusted to prevent unwanted oscillations that would occur if cubic splines were used. In addition, a weighting parameter gives sagging curvature to the distribution so that two-dimensional shockless distributions can be simulated.

The value Q1 at the nose should be set so that the resulting distribution lies below the initial analysis distribution along the lower surface and leading edge regions. Similarly, the value of Q4, the speed at the trailing edge, should be lower than the corresponding value computed in the analysis run. The two intermediate values, Q2 and Q3, define the size of the supersonic zone and the section lift. The prescribed wing surface can be thinned out near the supersonic zone by removing

material smoothly to produce a slight groove. The depth and extent of the groove are determined by the three parameters DSURF, PCS1 and PCS2.

When the optimization routine is used a sequence of calculations is performed in the inverse mode to determine the gradient of the wave drag with respect to the assigned Mach numbers that define the design distribution. After the gradient is obtained, a line search of five steps is performed to minimize the drag. This procedure can be adjusted by changing the parameters that appear in subroutines OPT and VAIOA.

The graphic output is produced in subroutines THREED and DRAGC and at the end of the main routine FL22INV. These programs have been written for the CDC 6600 at the Courant Institute of Mathematical Sciences and should be replaced by the plotting routines used at local installations or by dummy subroutines with the same names so that runs can be made without graphics.

Output appears in both printed and graphical form. All the input data is immediately printed as output so that it is easy to check the accuracy of the input.

At the beginning the coordinates defining the first span station are printed. If all the sections are similar only the coordinates of the leading edge, the chord and the twist are printed at the other stations. If the sections are different then the corresponding input profiles will be printed. The program prints the coordinates of the unfolded sections produced by the square root transformation at the root and the tip. A two-dimensional chart of the plane Y = 0 is printed giving values of an indicator IV which shows the properties of points in this coordinate surface. IV = 2 specifies a point on the wing; IV = 1 specifies a point on the trailing vortex sheet; IV = 0 specifies a point on the singular line X = 0; IV = -1 specifies a point adjacent to the wing or vortex sheet; and IV = -2 specifies a point beyond the wing or vortex sheet.

The iteration history is printed next. The maximum correction to the velocity potential and the maximum residual of the difference equations together with its i,j,k location are printed at every cycle. For an analysis run the lift coefficient CL, the wave drag coefficient CDW, two relaxation factors P10 and P20, a convergence factor BETA, and the number of supersonic points are printed at every iteration. For a design run, in addition to the correction and residual, the average difference between the computed and desired speeds and the corresponding maximum difference together with its i,k location are printed. The iteration cycles terminate after a given number of iterations or after a convergence criterion has been satisfied. A chart is then printed of the wave drag at the grid points (X,Z) of the wing surface. Supersonic points are

indicated by drag numbers IDRAG $\geq$ 0 and subsonic points are indicated by -1.

After this, results for each span section are displayed. The section lift, drag and moment coefficients are printed. For an analysis run the mapped wing surface and the Mach number distribution are displayed as a printer plot. For a design run, the prescribed surface and the computed surface are shown. The prescribed and computed Mach number distributions along the chord are shown for each span section. There are also Calcomp plots. The upper surface pressure distribution at each span section and the corresponding wing sections with markings that indicate the wave drag on shocks are plotted. In an analysis run the same plots occur for each mesh refinement.

The final printed results are the characteristics of the wing as a whole. These include the coefficients of lift, form drag, friction drag, total drag, the ratios of lift to form drag and lift to total drag, the pitching, rolling and yawing moments, and the wave drag.

For an analysis run, the program repeats the same sequence of calculations and output on successively refined meshes.

GLOSSARIES

The input files consist of sequences of pairs of cards. The first card of each pair gives the names of the parameters that appear on the data card that follows. Each data card contains up to eight parameters with 10 columns provided for each. The first input file described is needed for both analysis and design. If the code is used for analysis we are concerned with Tape 5 only and Card Pair 3 below does not exist. The input parameters are given in the order of their appearance on the input file. The input data is given in floating point format. The integer parameters are converted to integers in the code.

Glossary of Tape 5 Parameters

Card Pair 1:

NX — The number of mesh intervals in the direction of the chord. NX = 0 causes termination of the computation.

NY — The number of mesh intervals in the direction normal to the chord and span.

NZ	The number of mesh intervals in the span direction.
FPLOT	Controls the plots. FPLOT = 0. produces a print plot but no Calcomp plot. FPLOT $\geq$ 1. produces a print plot and a Calcomp plot.
XSCAL, PSCAL	These control the scales of the Calcomp plots. XSCAL = 0. scales each section plot to 5. PSCAL = 0. scales the pressure plots to 1. per inch.
FCONT	Determines the manner of starting the program. FCONT = 0. begins the calculation at iteration zero. FCONT = 1. continues the calculation from a previous calculation. For a design run the flow data (velocity potential and circulation) from an analysis run are read in on Tape 7 and used for initialization. The iteration count starts from zero for a design run.
FSWEEP	An indicator which selects the subroutine YSWEEP used to solve the finite difference equations for the reduced potential G. FSWEEP = 0. selects a vectorized YSWEEP subroutine. FSWEEP = 1. selects an unvectorized YSWEEP subroutine.

Card Pair 2:

MIT	The maximum number of iteration cycles which will be computed.
COV	The desired accuracy. If the maximum correction is less than COV the calculation terminates or proceeds to a finer mesh.
P10	The subsonic relaxation factor for the velocity potential. P10 lies between 1. and 2. and should be increased linearly toward 2. with mesh refinement.

P20 — The supersonic relaxation factor for the velocity potential.

$P20 \leq 1.$ Recommended value 1.

P30 — The relaxation factor for the circulation. Recommended value 1.

BETA — The damping factor which controls the amount of added ϕ_{st}. Recommended value between 0. and 0.25.

FHALF — Determines whether the mesh will be refined.

FHALF = 0. terminates the computation after MIT iterations or after convergence.

$FHALF \neq 0.$ halves the mesh after MIT cycles have been run on the crude mesh. An additional Card Pair 2 is required for each mesh refinement. The value FHALF = 0. appears on the last mesh refinement card.

NDES — Gives the number of surface modifications to be calculated.

$NDES \leq 0.$ calls for an analysis run.

NDES > 0. makes a design calculation with NDES surface modifications. MIT cycles of line relaxation are performed after each surface modification. No mesh refinements are made after the NDES cycles are completed. MIT = 1. with $NDES \geq 100.$ is recommended. If NDES > 0. an additional Card Pair 3 is needed at this point.

Card Pair 3:

TSTEP — TSTEP times the mesh increment in the X direction is the time step defining the free boundary iteration. The recommended value is 0.03.

F00 — The coefficient multiplying the first order time derivative in the free boundary equation. This term dominates for subsonic flow. Recommended value F00 = 0.16.

F10 — The coefficient of the second derivative with respect to time and the X coordinate in the free boundary equation. This term controls convergence for supersonic flow. Recommended value F10 = -1.

FOPT	FOPT $\leq$ 0. indicates a regular inverse run. FOPT > 0. invokes the optimization procedure.

Card Pair 4:

FMACH	The free stream Mach number.
YAW	The yaw angle of the wing in degrees.
ALPHA	The angle of attack in degrees.
CD0	The estimated drag due to skin friction. This can be read in and added to the drag calculated by the program to give the total drag.

Card Pair 5:

ZSYM	Indicates whether an isolated wing or a wing on a wall is being considered. ZSYM = 0. specifies an isolated wing at a prescribed yaw angle; obsolescent. ZSYM = 1. specifies a swept wing on a wall.
NC	The number of span stations from the wing root to the tip at which the wing section is defined if ZSYM = 1. For ZSYM = 0. the span stations are distributed from the leading to the trailing tip. The wing sections are each defined on subsequent cards.
SWEEP1	Sweep of the singular line at the wing root if ZSYM = 1. or at the leading tip if ZSYM = 0.
SWEEP2	Sweep of the singular line at the tip. SWEEP1 and SWEEP2 are used as the end conditions for the spline fit for the X coordinates of the singular line.
SWEEP3	Sweep of the singular line in the far field.
DIHED1	Dihedral angle of the singular line at the wing root if ZSYM = 1. or at the leading tip if ZSYM = 0.
DIHED2	Dihedral angle of the singular line at the wing tip. DIHED1 and DIHED2 are used as the end conditions for the spline fit of the Y coordinates of the singular line.
DIHED	Dihedral angle of the singular line in the far field.

Card Pair 6:

Z	Span location of the section.
XLE, YLE	X and Y coordinates of the leading edge.
CHORD	The local chord value by which the profile coordinates are scaled.
THICK	Modifies the section thickness. The Y coordinates are multiplied by THICK.
ALPHA	The angle through which a section is rotated to introduce twist.
FSEC	Indicates whether or not the geometry for a new profile is supplied. FSEC = 0. means the section is obtained by scaling the profile used at the previous span section according to the parameters CHORD, THICK, and ALPHA. No further cards are read for this span station and the next card is the title card for the next span station, if any. FSEC = 1. means the coordinates for a new profile are to be read from the data cards that follow.

Card Pair 7:

YSYM	Indicates the type or profile. YSYM = 0. means the data supplied are for a cambered profile. Coordinates are given for the upper and lower surfaces, each ordered from nose to tail with the leading edge included in both surfaces. YSYM = 1. means the data supplied are for a symmetric profile. A table of coordinates is read in for the upper surface only.
NU	The number of upper surface coordinates.
NL	The number of lower surface coordinates. For YSYM = 1., NL = NU.

Card Pair 8:

TRAIL	The included angle at the trailing edge in degrees. If the profile is open then TRAIL is the difference between the upper and lower trailing edge angles.

SLOPT — The slope of the mean camber line at the trailing edge. This is used to continue the coordinate surface, assumed to contain the vortex sheet, smoothly off the trailing edge.

XSING, YSING — The coordinates of the singular point inside the nose about which the square root transformation is applied to generate parabolic coordinates. This point should be located as symmetrically as possible between the upper and lower surfaces at a distance from the nose roughly proportional to the leading edge radius. The coordinates of the mapped profile in the output will show if this point has been located correctly. The coordinates of the singular point are chosen relative to the profile coordinates supplied in the cards which follow.

Card Pair 9: (Upper surface coordinates.)

X, Y — The coordinates of the upper surface. They appear, a pair on each card, from leading edge to trailing edge. The format is (2F10.6).

Card Pair 10: (Lower surface coordinates.)

X, Y — The coordinates of the lower surface from leading edge to trailing edge. The leading edge point of the upper surface is the same as the leading edge point of the lower surface. The trailing edge points are different if the profile has an open tail.

Card Pairs 11,12,...: (Geometry at other span stations.)

These cards are like Card Pair 6, which defines Z, XLE, YLE, CHORD, THICK, ALPHA and FSEC for each section. The number of such cards depends on the number of span stations, NC. If FSEC = 0. new coordinates X,Y are not needed to define the profile.

The last card pair:

The card which terminates the run is a repeat of Card Pair 1 with all the data set equal to zero.

Glossary of Tape 9 Parameters

Card Pair 1:

NQSTA	The number of span stations at which the design distribution is defined from wing root to tip.

Card Pair 2: (Card Pairs 2 and 3 are repeated NQSTA times.)

ZQSTA	The Z coordinate of the span section at which the design distribution is given.
PCQ1	The location of the first specified value Q1 of the speed, expressed as a fraction of the local chord.
PCQ2	Location of the second specified value Q2.
PCQ3	Location of the third specified value Q3. (PCQ4 = 1 because the speed Q_4 is always prescribed at the trailing edge.)
Q1	The first prescribed Mach number near the leading edge used in spline fitting the design distribution at each section.
Q2	The prescribed Mach number near the front of the supersonic zone.
Q3	The prescribed Mach number near the rear of the supersonic zone.
Q4	The prescribed Mach number at the trailing edge.

Card Pair 3: (These parameters are used to define the groove for each span station.)

PCS1	Location of the left edge of the groove expressed as a fraction of the local chord. The groove is assumed to be on the upper surface.
PCS2	Location of the right edge of the groove expressed as a fraction of the local chord.
DSURF	The maximum depth of the groove expressed in units used in the computational domain.

REFERENCES

1. F. Bauer, P. Garabedian and D. Korn, A Theory of Supercritical Wing Sections, with Computer Programs and Examples, Lecture Notes in Economics and Mathematical Systems 66, Springer-Verlag, New York, 1972.

2. F. Bauer, P. Garabedian, D. Korn and A. Jameson, Supercritical Wing Sections II, A Handbook, Lecture Notes in Economics and Mathematical Systems 108, Springer-Verlag, New York, 1975.

3. F. Bauer, P. Garabedian and D. Korn, Supercritical Wing Sections III, Lecture Notes in Economics and Mathematical Systems 150, Springer-Verlag, New York, 1977.

4. A. Bayliss, M. Gunzburger and E. Turkel, "Boundary conditions for the numerical solution of elliptic equations in exterior domains," SIAM J. Appl. Math. 42, 430-451 (1982).

5. O. Betancourt and P. Garabedian, "Numerical analysis of equilibria with islands in magnetohydrodynamics," Comm. Pure Appl. Math. 35, 365-378 (1982).

6. P. Garabedian and G. McFadden, "Design of supercritical swept wings," A.I.A.A. Journal 20, 289-291 (1982).

7. P. Garabedian and G. McFadden, "Computational fluid dynamics of airfoils and wings," in: Transonic, Shock, and Multidimensional Flows: Advances in Scientific Computing, Academic Press, New York, 1982.

8. A. Jameson and D. Caughey, "Numerical calculation of the transonic flow past a swept wing," NASA CR-153297, 1977.

9. E. Murman and J. Cole, "Calculation of plane steady transonic flows," A.I.A.A. Journal 9, 114-121 (1971).

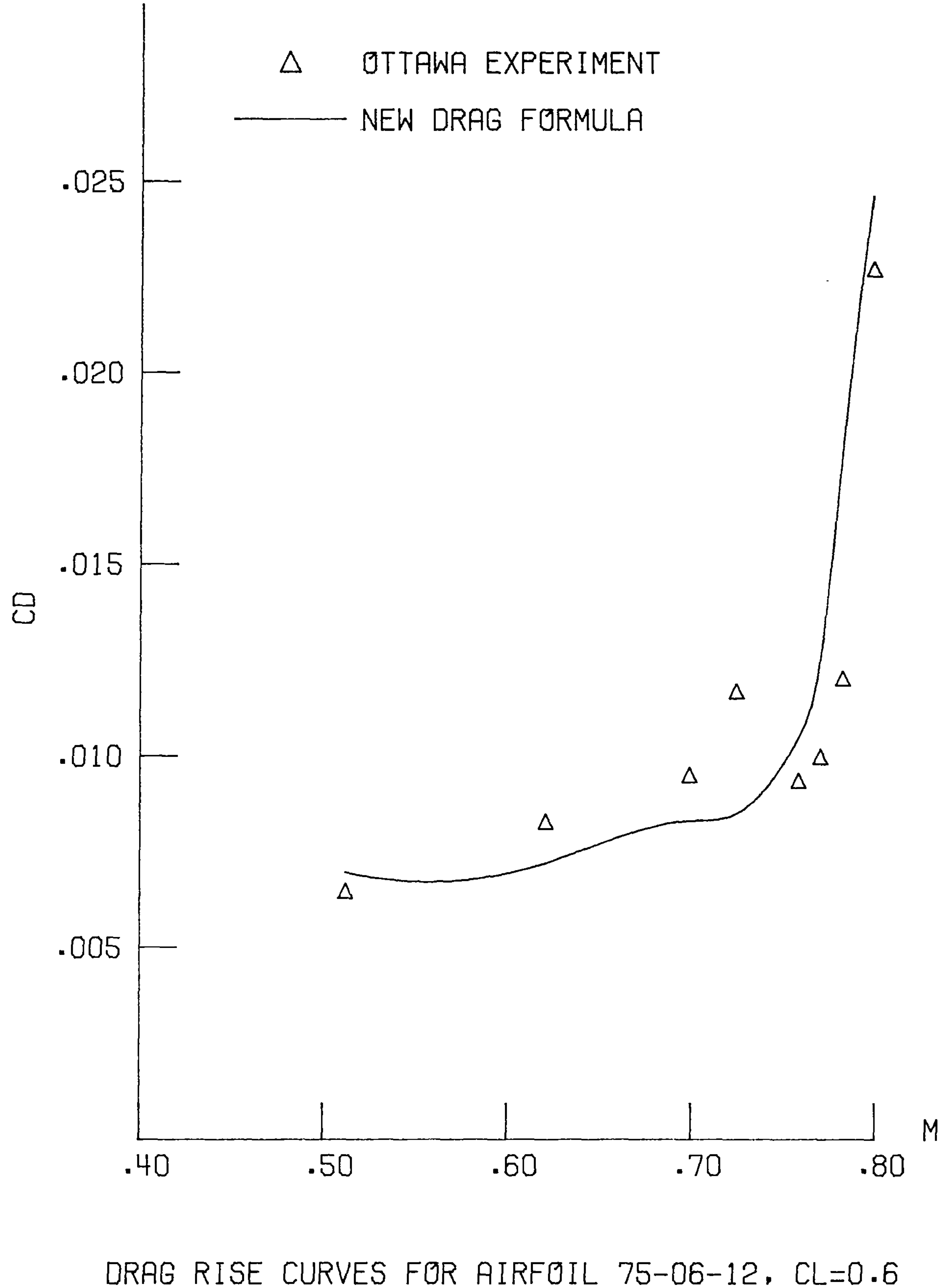

Fig. 1. Theoretical and experimental drag rise curves.

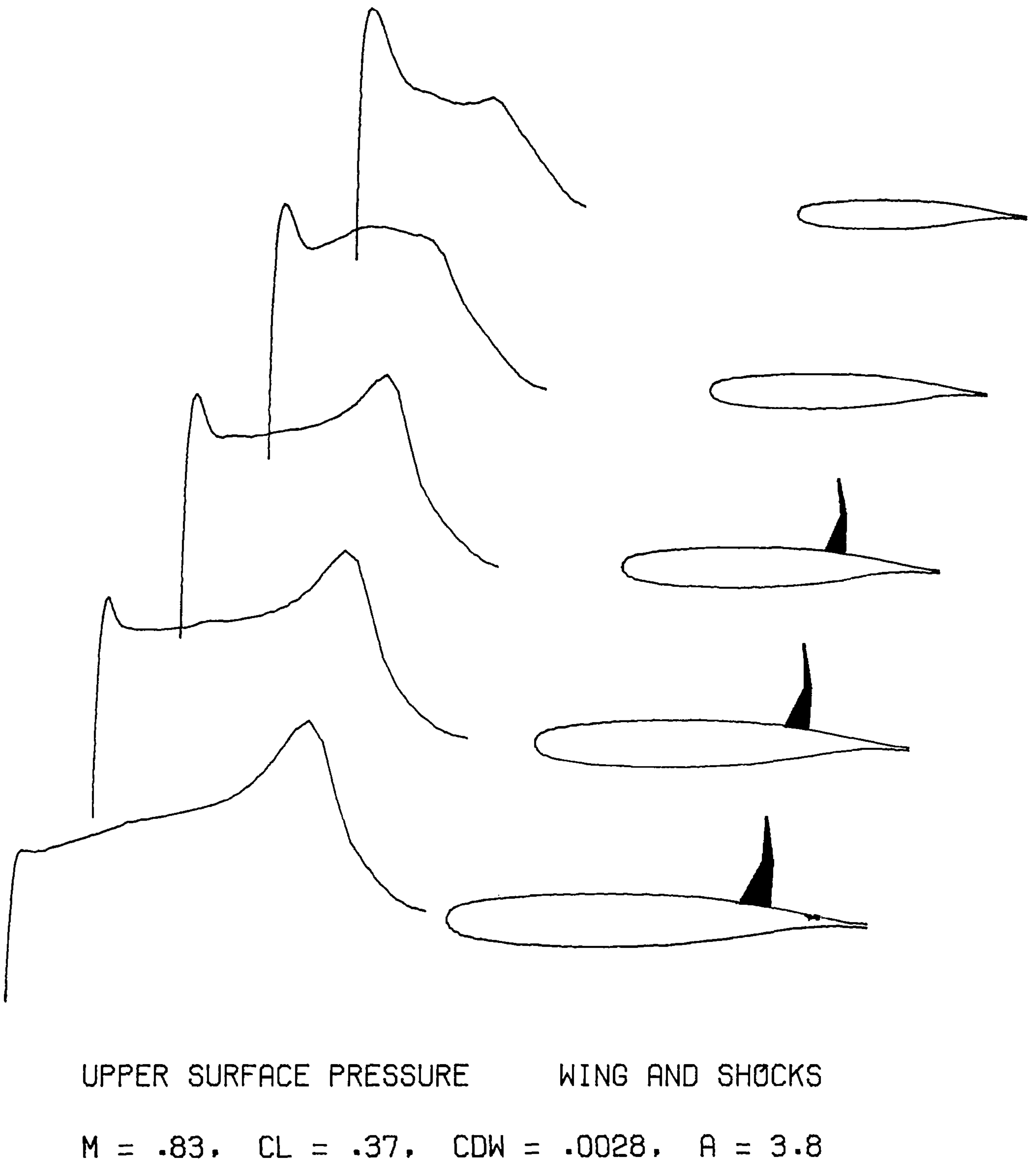

Fig. 2. Calcomp plot for analysis run preceding design.

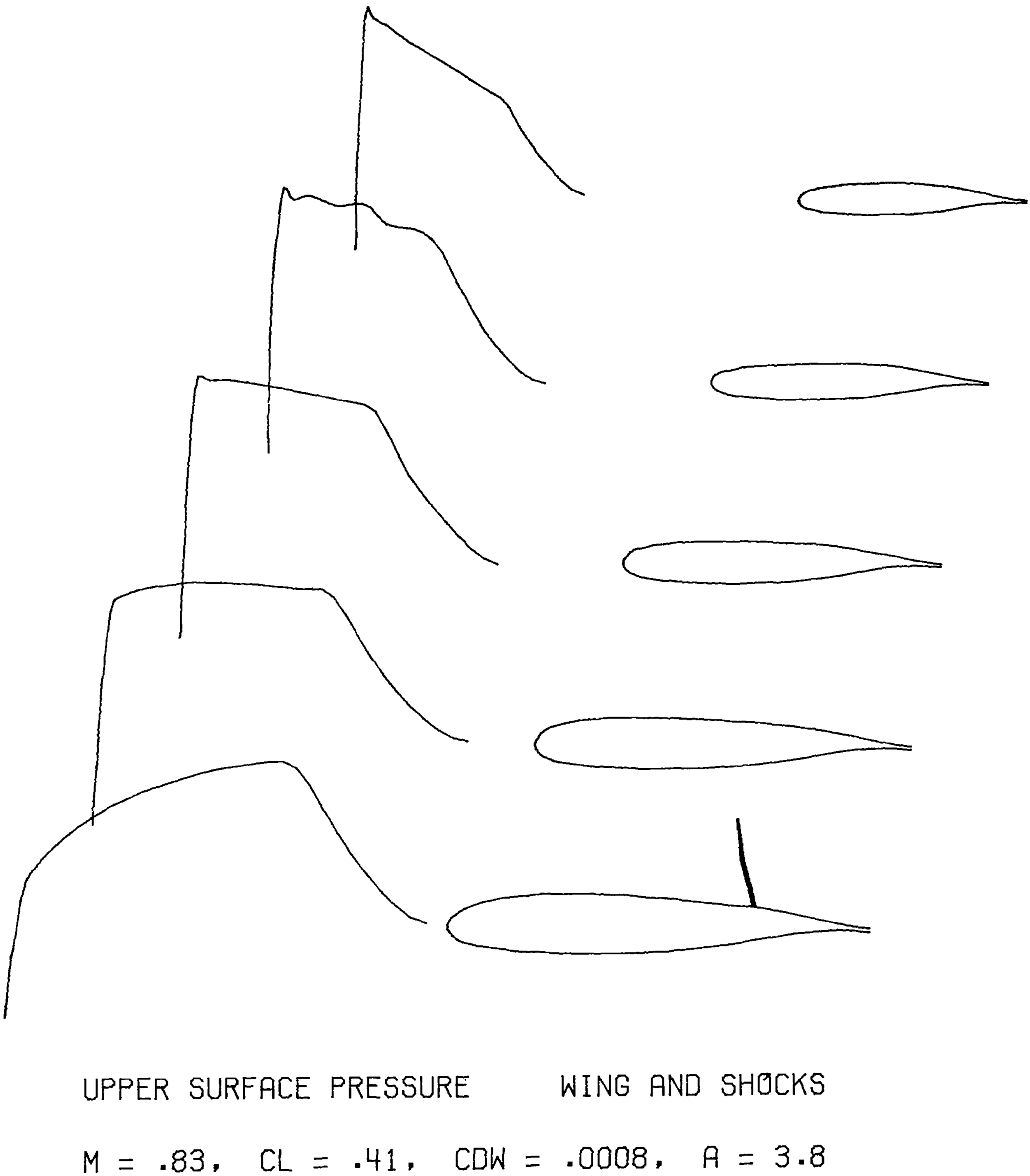

Fig. 3. Calcomp plot for the sample design run.

LISTING OF A SAMPLE RUN

```
 FNX            FNY            FNZ            FPLOT
 .12800E+03     .10000E+02     .12000E+02    0.

     XSCAL          PSCAL          FCONT          FSWEEP
     .56000E+01    -.50000E+00     .10000E+01     .10000E+01

 FIT(NM)        COVO(NM)       P10(NM)        P20(NM)
 .10000E+01     .10000E-06     .17200E+01     .10000E+01

     P30(NM)        BETAO(NM)      FHALF(NM)      FDES(NM)
     .10000E+01     .10000E+00    0.              .10000E+03

 TSTEP          FOO            F10            F11
 .30000E-01     .16000E+00    -.10000E+01    0.

     FOPT
    0.

 FMACH          YA              AL             CDO
  .83000E+00     0.               .10000E+01     0.

 ZSYM           FNC            SWEEP1         SWEEP2
 .10000E+01     .60000E+01     .30000E+02     .30000E+02

     SWEEP          DIHED1         DIHED2         DIHED
     .30000E+02    0.             0.             0.

 ZS(K)          XL             YL
0.             0.             0.

     CHORD          THICK          AL             FSEC
     .67370E+00     .10000E+01    0.             0.

 YSYM           FNU            FNL
0.              .47000E+02     .35000E+02

 TRL            SLT            XSING          YSING
  .1577E+01     -.1000E+00      .1000E-01      .1403E-01
```

UPPER SURFACE		LOWER SURFACE	
XP(I)	YP(I)	XP(I)	YP(I)
-.28010E-02	.22664E-01	-.28010E-02	.22664E-01
.46130E-02	.35103E-01	0.	0.
.19370E-01	.46090E-01	.80950E-02	-.98270E-02
.26231E-01	.49082E-01	.19227E-01	-.17255E-01
.35059E-01	.52122E-01	.33397E-01	-.23432E-01
.45415E-01	.54992E-01	.49561E-01	-.28572E-01
.57105E-01	.57681E-01	.67610E-01	-.32891E-01
.69832E-01	.60167E-01	.87596E-01	-.36589E-01
.83659E-01	.62497E-01	.10980E+00	-.39868E-01
.98782E-01	.64716E-01	.13392E+00	-.42776E-01
.11506E+00	.66806E-01	.15988E+00	-.45339E-01
.13213E+00	.68731E-01	.18742E+00	-.47551E-01
.14999E+00	.70501E-01	.21663E+00	-.49431E-01
.16893E+00	.72148E-01	.24724E+00	-.50965E-01
.18896E+00	.73667E-01	.27937E+00	-.52158E-01
.20982E+00	.75033E-01	.31264E+00	-.52800E-01
.23146E+00	.76243E-01	.34718E+00	-.53145E-01
.25400E+00	.77299E-01	.38259E+00	-.53076E-01
.27723E+00	.78188E-01	.41913E+00	-.52537E-01
.30076E+00	.78895E-01	.45636E+00	-.51470E-01
.32450E+00	.79252E-01	.49472E+00	-.49808E-01
.34853E+00	.79521E-01	.53380E+00	-.47365E-01
.37262E+00	.79618E-01	.57433E+00	-.43957E-01
.39645E+00	.79550E-01	.61612E+00	-.39376E-01
.41996E+00	.79320E-01	.66026E+00	-.33304E-01
.44315E+00	.78932E-01	.70633E+00	-.25802E-01
.46569E+00	.78400E-01	.75504E+00	-.17281E-01
.48715E+00	.77747E-01	.80455E+00	-.92630E-02
.50725E+00	.76996E-01	.85410E+00	-.35060E-02
.52567E+00	.76184E-01	.89932E+00	-.10920E-02
.54206E+00	.75346E-01	.93856E+00	-.13510E-02
.55629E+00	.74526E-01	.96789E+00	-.27310E-02
.56854E+00	.73734E-01	.98810E+00	-.41570E-02
.57908E+00	.72982E-01	.99652E+00	-.49110E-02
.58821E+00	.72270E-01	.99733E+00	-.50020E-02
.59637E+00	.71577E-01		
.60433E+00	.70852E-01		
.61266E+00	.70035E-01		
.67205E+00	.62571E-01		
.73993E+00	.51620E-01		
.80171E+00	.39381E-01		
.86178E+00	.26501E-01		
.91109E+00	.16163E-01		
.95185E+00	.87060E-02		
.97857E+00	.47780E-02		
.99411E+00	.29260E-02		
.99673E+00	.26110E-02		

```
SECTION DEFINITION AT Z =     0.00000
 XLE              YLE             CHORD     THICKNESS RATIO     ALPHA
 0.0000           0.0000            .6737          1.0000        0.0000

ZS(K)           XL              YL
.20000E+00     .11500E+00    0.

    CHORD           THICK          AL              FSEC
    .61470E+00     .10000E+01    0.              0.

SECTION DEFINITION AT Z =      .20000
 XLE              YLE             CHORD     THICKNESS RATIO     ALPHA
  .1150           0.0000            .6147          1.0000        0.0000

ZS(K)           XL              YL
.40000E+00     .23000E+00    0.

    CHORD           THICK          AL              FSEC
    .55580E+00     .10000E+01    0.              0.

SECTION DEFINITION AT Z =      .40000
 XLE              YLE             CHORD     THICKNESS RATIO     ALPHA
  .2300           0.0000            .5558          1.0000        0.0000

ZS(K)           XL              YL
.60000E+00     .34500E+00    0.

    CHORD           THICK          AL              FSEC
    .49680E+00     .10000E+01    0.              0.

SECTION DEFINITION AT Z =      .60000
 XLE              YLE             CHORD     THICKNESS RATIO     ALPHA
  .3450           0.0000            .4968          1.0000        0.0000

ZS(K)           XL              YL
.80000E+00     .46000E+00    0.

    CHORD           THICK          AL              FSEC
    .43790E+00     .10000E+01    0.              0.

SECTION DEFINITION AT Z =      .80000
 XLE              YLE             CHORD     THICKNESS RATIO     ALPHA
  .4600           0.0000            .4379          1.0000        0.0000

ZS(K)           XL              YL
.10000E+01     .57500E+00    0.

    CHORD           THICK          AL              FSEC
    .37890E+00     .10000E+01    0.              0.

SECTION DEFINITION AT Z =     1.00000
 XLE              YLE             CHORD     THICKNESS RATIO     ALPHA
  .5750           0.0000            .3789          1.0000        0.0000
```

PARAMETERS TO DEFINE THE ASSIGNED DESIGN MACH NUMBER DISTRIBUTION

K	Z	PCM1	PCM2	PCM3	M1	M2	M3	M4
1	0.000	.065	.220	.830	.420	.850	1.240	.670
2	.250	.055	.220	.800	.440	1.140	1.190	.680
3	.500	.055	.220	.780	.440	1.270	1.210	.670
4	.875	.065	.220	.780	.480	1.370	1.120	.660
5	1.000	.065	.200	.820	.500	1.345	1.000	.700

PCX1	PCX2	DSURF
.500	.900	.002
.400	.900	.002
.400	.900	.002
.150	.900	0.000
.150	.900	0.000

INDICATION IV(I,K) OF WING AND VORTEX SHEET IN PLANE Y=0
((IV(I,K),K=K1,K2),I=2,NX)

1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
2	1	1	1	1	1	1	1	1	-1	-2	-2
2	1	1	1	1	1	1	1	1	-1	-2	-2
2	2	1	1	1	1	1	1	1	-1	-2	-2
2	2	2	1	1	1	1	1	1	-1	-2	-2
2	2	2	1	1	1	1	1	1	-1	-2	-2
2	2	2	2	1	1	1	1	1	-1	-2	-2
2	2	2	2	1	1	1	1	1	-1	-2	-2
2	2	2	2	2	1	1	1	1	-1	-2	-2
2	2	2	2	2	2	1	1	1	-1	-2	-2
2	2	2	2	2	2	1	1	1	-1	-2	-2
2	2	2	2	2	2	2	1	1	-1	-2	-2
2	2	2	2	2	2	2	1	1	-1	-2	-2
2	2	2	2	2	2	2	2	1	-1	-2	-2
2	2	2	2	2	2	2	2	1	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2

2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	0	0	0
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	2	-1	-2	-2
2	2	2	2	2	2	2	2	1	-1	-2	-2
2	2	2	2	2	2	2	2	1	-1	-2	-2
2	2	2	2	2	2	2	1	1	-1	-2	-2
2	2	2	2	2	2	2	1	1	-1	-2	-2
2	2	2	2	2	2	1	1	1	-1	-2	-2

2	2	2	2	2	2	1	1	1	-1	-2	-2
2	2	2	2	2	1	1	1	1	-1	-2	-2
2	2	2	2	1	1	1	1	1	-1	-2	-2
2	2	2	2	1	1	1	1	1	-1	-2	-2
2	2	2	1	1	1	1	1	1	-1	-2	-2
2	2	2	1	1	1	1	1	1	-1	-2	-2
2	2	1	1	1	1	1	1	1	-1	-2	-2
2	1	1	1	1	1	1	1	1	-1	-2	-2
2	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2
1	1	1	1	1	1	1	1	1	-1	-2	-2

NORMAL CELL DISTRIBUTION IN SQUARE ROOT PLANE

Y
.56695
.44721
.35909
.28868
.22917
.17678
.12910
.08452
.04181
0.00000

SCALE FACTOR	POWER LAW
.50000	.50000

SPANWISE CELL DISTRIBUTION AND SINGULAR LINE

Z	X SING	Y SING	XZ
0.00000	.00862	-.00582	.57735
.12500	.08018	-.00556	.56934
.25000	.15138	-.00516	.57083
.37500	.22282	-.00486	.57173
.50000	.29424	-.00454	.57106
.62500	.36565	-.00423	.57173
.75000	.43709	-.00393	.57083
.87500	.50829	-.00353	.56934
1.00000	.57985	-.00327	.57735
1.12677	.65304	-.00327	.57735
1.26516	.73294	-.00327	.57735
1.43301	.82985	-.00327	.57735

YZ	XZZ	YZZ
.00000	-.10644	.04426
.00333	-.02173	.00902
.00271	.01938	-.00807
.00233	-.00491	.00206
.00261	-.00000	.00000
.00233	.00491	-.00206
.00271	-.01938	.00807
.00333	.02173	-.00902
.00000	.10644	-.04426
0.00000	0.00000	0.00000
0.00000	0.00000	0.00000
0.00000	0.00000	0.00000

TIP LOCATION	POWER LAW
.57143	.50000

ITERATIVE SOLUTION
STRIP WIDTH FOR HORIZONTAL LINE RELAXATION
1.00000

NX	NY	NZ
128	10	12

COMPUTING TIME 47.497 SECONDS

MACH NO	YAW	ANG OF ATTACK
.83000	0.00000	1.00000

ITERATION	RESIDUAL	I	J	K	AVERAGE Q	DRAG
1	.32652E-05	128	10	6	0.	.0028
2	.22339E-04	128	10	5	.24099E+00	.0028
3	-.33525E-04	110	11	4	.23144E+00	.0028
4	-.66452E-04	111	11	3	.22337E+00	.0027
5	-.65743E-04	111	11	3	.21972E+00	.0026
6	-.61413E-04	111	11	3	.21434E+00	.0025
7	-.55446E-04	111	11	3	.20744E+00	.0024
8	-.49685E-04	111	11	3	.20334E+00	.0023
9	-.42556E-04	111	11	3	.19688E+00	.0022
10	-.39147E-04	111	11	3	.19178E+00	.0022
11	.39657E-04	106	11	3	.18661E+00	.0021
12	.40238E-04	106	11	3	.18115E+00	.0020
13	.40815E-04	106	11	3	.17402E+00	.0020
14	.41442E-04	106	11	3	.16688E+00	.0019
15	.41479E-04	106	11	3	.16052E+00	.0019
16	.42238E-04	118	11	3	.15602E+00	.0018
17	.44582E-04	118	11	3	.15039E+00	.0018
18	.46007E-04	118	11	3	.14704E+00	.0017
19	.47812E-04	118	11	3	.14158E+00	.0017
20	.48790E-04	118	11	3	.13557E+00	.0016
21	.49760E-04	118	11	3	.13093E+00	.0016
22	.50483E-04	118	11	3	.12642E+00	.0016
23	.50944E-04	118	11	3	.12177E+00	.0015
24	.51533E-04	118	11	3	.11737E+00	.0015
25	.51600E-04	118	11	3	.11316E+00	.0015
26	.51489E-04	118	11	3	.10941E+00	.0015
27	.51818E-04	118	11	3	.10650E+00	.0015
28	.51418E-04	118	11	3	.10299E+00	.0014
29	.51416E-04	118	11	3	.99838E-01	.0014
30	.51081E-04	118	11	3	.96995E-01	.0014
31	.50849E-04	118	11	3	.93886E-01	.0014
32	.50622E-04	118	11	3	.91089E-01	.0014
33	.50489E-04	118	11	3	.88611E-01	.0014
34	.50499E-04	118	11	3	.86250E-01	.0014
35	.50617E-04	118	11	3	.82413E-01	.0014
36	.50744E-04	118	11	3	.79798E-01	.0014
37	.50766E-04	118	11	3	.77769E-01	.0014
38	.50833E-04	118	11	3	.76239E-01	.0014
39	.50722E-04	118	11	3	.74098E-01	.0014
40	.50504E-04	118	11	3	.71992E-01	.0014
41	.50140E-04	118	11	3	.70111E-01	.0014
42	.49725E-04	118	11	3	.68546E-01	.0014
43	.49258E-04	118	11	3	.66656E-01	.0014
44	.48772E-04	118	11	3	.64673E-01	.0014
45	.48218E-04	118	11	3	.63204E-01	.0014
46	.47602E-04	118	11	3	.62074E-01	.0014

47	.46976E-04	118	11	3	.60766E-01	.0014
48	.46369E-04	118	11	3	.59644E-01	.0014
49	.45817E-04	118	11	3	.58050E-01	.0014
50	.45222E-04	118	11	3	.56752E-01	.0014
51	.44648E-04	118	11	3	.55776E-01	.0014
52	.44038E-04	118	11	3	.54843E-01	.0014
53	.43466E-04	118	11	3	.53823E-01	.0014
54	.42837E-04	118	11	3	.52818E-01	.0013
55	.42229E-04	118	11	3	.51770E-01	.0013
56	.41618E-04	118	11	3	.50704E-01	.0013
57	.41004E-04	118	11	3	.49794E-01	.0013
58	.40368E-04	118	11	3	.48961E-01	.0013
59	.39772E-04	118	11	3	.47793E-01	.0013
60	.39209E-04	118	11	3	.46724E-01	.0013
61	.38655E-04	118	11	3	.45672E-01	.0013
62	.38039E-04	118	11	3	.45067E-01	.0012
63	.37509E-04	118	11	3	.44301E-01	.0012
64	.36847E-04	118	11	3	.43454E-01	.0012
65	.36252E-04	118	11	3	.42829E-01	.0012
66	.35627E-04	118	11	3	.42009E-01	.0012
67	.35008E-04	118	11	3	.41134E-01	.0012
68	.34363E-04	118	11	3	.40542E-01	.0012
69	.33749E-04	118	11	3	.39980E-01	.0011
70	.33131E-04	118	11	3	.39322E-01	.0011
71	.32486E-04	118	11	3	.38786E-01	.0011
72	.31794E-04	118	11	3	.37994E-01	.0011
73	.31138E-04	118	11	3	.37112E-01	.0011
74	.30488E-04	118	11	3	.36548E-01	.0011
75	.29836E-04	118	11	3	.35976E-01	.0011
76	.29193E-04	118	11	3	.35314E-01	.0010
77	.28521E-04	118	11	3	.34710E-01	.0010
78	.27883E-04	118	11	3	.34138E-01	.0010
79	.27184E-04	118	11	3	.33668E-01	.0010
80	.26540E-04	118	11	3	.33157E-01	.0010
81	.25852E-04	118	11	3	.32543E-01	.0010
82	.25194E-04	118	11	3	.32108E-01	.0010
83	.24503E-04	118	11	3	.31485E-01	.0009
84	.23841E-04	118	11	3	.31071E-01	.0009
85	.23131E-04	118	11	3	.30416E-01	.0009
86	.22452E-04	118	11	3	.29559E-01	.0009
87	.21766E-04	118	11	3	.29395E-01	.0009
88	.21059E-04	118	11	3	.29200E-01	.0009
89	.20368E-04	118	11	3	.28790E-01	.0009
90	.19682E-04	118	11	3	.28600E-01	.0009
91	.18975E-04	118	11	3	.28460E-01	.0009
92	.18292E-04	118	11	3	.28356E-01	.0008
93	.17604E-04	118	11	3	.28030E-01	.0008
94	.16921E-04	118	11	3	.27879E-01	.0008
95	.16242E-04	118	11	3	.27734E-01	.0008
96	.15568E-04	118	11	3	.27594E-01	.0008
97	.14893E-04	118	11	3	.27635E-01	.0008
98	.14235E-04	118	11	3	.27428E-01	.0008
99	.13593E-04	118	11	3	.27197E-01	.0008
100	.12961E-04	118	11	3	.26965E-01	.0008

MAX RESIDAL 1	MAX RESIDAL 2	WORK	REDUCTN/CYCLE
.3265E-05	.1296E-04	99.0000	1.0140

COMPUTING TIME 672.962 SECONDS

WAVE DRAG = .00078
PRINTOUT OF IDRAG(I,K)

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	0	-1
-1	-1	-1	-1	-1	-1	0	0	0
-1	-1	-1	-1	-1	-1	0	0	0
-1	-1	-1	-1	0	0	0	0	0
-1	-1	-1	0	0	0	0	0	0
-1	-1	0	0	0	0	0	0	0
-1	-1	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	10	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	6	-1
0	0	0	0	0	0	4	23	-1
0	0	0	0	0	0	25	-1	-1
0	0	0	0	0	10	22	-1	-1
0	0	0	0	13	57	-1	-1	-1
0	0	0	0	75	19	-1	-1	-1
0	0	0	15	25	-1	-1	-1	-1
0	0	0	66	-1	-1	-1	-1	-1
0	0	23	21	-1	-1	-1	-1	
0	0	59	-1	-1	-1	-1	-1	
0	66	20	-1	-1	-1	-1		
0	149	-1	-1	-1	-1	-1		
19	17	-1	-1	-1	-1			
90	-1	-1	-1	-1	-1			
47	-1	-1	-1	-1				
-1	-1	-1	-1					
-1	-1	-1	-1					
-1	-1	-1						
-1	-1	-1						
-1	-1							
-1								
-1								

WING CHARACTERISTICS

MACH NO	YAW	ANG OF ATTACK	
.83000	0.00000	1.00000	
CL	CD FORM	CD FRICTION	CD
.40535	.01729	0.00000	.01729
	L/D FORM	L/D	
	23.43966	23.43966	
CD WAVE			
.00078			
CM PITCH	CM ROLL	CM YAW	AWING
-.39938	.37631	.00116	.52764

LISTING OF THE CODE

```
      PROGRAM FL22INV(INPUT=512,OUTPUT=512,TAPE5=INPUT,TAPE6=OUTPUT,TAPE
     114=0,TAPE15=0,TAPE16=0,TAPE1=0,TAPE7=512,TAPE8=512,TAPE11=0,TAPE9=
     264,TAPE10=512)
C     MAIN ROUTINE WHICH CONTROLS THE COMPUTATIONAL PROCEDURE.
C     G IS REDUCED VELOCITY POTENTIAL
      COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
     1,ITE2(15),A0(129),A1(129),A2(129),A3(129),B0(12),B1(12),B2(12),B3(
     212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
     3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
     4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
     515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
     6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,F00,F01,F10,F11,
     7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
      COMMON /FLO/ STRIP,P1,P2,P3,BETA,FR,IR,JR,KR,DG,IG,JG,KG,NS,FSWEEP
      DIMENSION XS(200,11), YS(200,11), ZS(11), XLE(11), YLE(11), SLOPT(
     111), TRAIL(11), NP(11), E1(11), E2(11), E3(11), E4(11), E5(11), XP
     2(241), YP(241), D1(241), D2(241), D3(241), X(129), Y(129), SV(129)
     3, SM(129), CP(129), CHORD(15), SCL(15), SCD(15), SCM(15), FIT(3),
     4COVO(3), P10(3), P20(3), P30(3), BETAO(3), STRIPO(3), FHALF(3), RE
     5S(501), COUNT(501), UC(129), VC(129), WC(129), FDES(3), CLO(11), C
     6LPRE(15), ALFO(15)
      DIMENSION DESC(10), LABEL(10), NPARAM(30), TITLE(10)
      COMMON /DIM/ NX1,NY1,NZ1,FDIM
      ND=200
      NE=129
      IREAD=5
      IWRIT=6
      KPLOT=0
      IPLOT=1
      ISTOP=2
      JO=0
      NF1=1
      REWIND 7
      RAD=57.2957795130823
   10 WRITE (IWRIT,670)
      WRITE (IWRIT,390)
      READ (IREAD,660) TITLE
      WRITE (IWRIT,700) TITLE
      READ (IREAD,660) DESC
      READ (IREAD,650) FNX,FNY,FNZ,FPLOT,XSCAL,PSCAL,FCONT,FSWEEP
      WRITE (IWRIT,780) FNX,FNY,FNZ,FPLOT,XSCAL,PSCAL,FCONT,FSWEEP
      NX=FNX
      NY=FNY
      NZ=FNZ
      IF (NX.LT.1) GO TO 380
      KPLOT=ABS(FPLOT)
```

```
      READ (IREAD,660) DESC
      WRITE (IWRIT,790)
      NM=0
   20 NM=NM+1
      READ (IREAD,650) FIT(NM),COVO(NM),P10(NM),P20(NM),P30(NM),BETAO(NM
     1),FHALF(NM),FDES(NM)
      STRIPO(NM)=1.0
      WRITE (IWRIT,690) FIT(NM),COVO(NM),P10(NM),P20(NM),P30(NM),BETAO(N
     1M),FHALF(NM),FDES(NM)
      IF (FHALF(NM).NE.0..AND.NM.LT.3) GO TO 20
      IF (FDES(1).LE.0.) GO TO 30
      READ (IREAD,660) DESC
      READ (IREAD,650) TSTEP,F00,F10,F11,FOPT
      WRITE (IWRIT,400) DESC
      WRITE (IWRIT,410) TSTEP,F00,F10,F11,FOPT
   30 CONTINUE
      NMESH=NM
      FHALF(3)=0.
      READ (IREAD,660) DESC
      READ (IREAD,650) FMACH,YA,AL,CDO,CLOPT,RCL,SREF
      WRITE (IWRIT,800) FMACH,YA,AL,CDO,SREF
      YAW=YA/RAD
      ALPHA=AL/RAD
      CALL GEOM (ND,NC,NP,ZS,XS,YS,XLE,YLE,SLOPT,TRAIL,XP,YP,SWEEP1,SWEE
     1P2,SWEEP,DIHED1,DIHED2,DIHED,XTEO,CHORDO,ZTIP,ISYMO,KSYM,CLO)
      ISYM=ISYMO
      IF (ALPHA.NE.0.) ISYM=0
      IF (KSYM.NE.0) YAW=0.
      CYAW=COS(YAW)
      SYAW=SIN(YAW)
      CA=CYAW*COS(ALPHA)
      SA=CYAW*SIN(ALPHA)
      IF (FDES(1).GT.0.) CALL READQS (NQSTA,ZQSTA,PCQ1,PCQ2,PCQ3,QQ1,QQ2
     1,QQ3,QQ4,PCS1,PCS2,DSURF,FMACH)
      IF (FCONT.LT.1.) GO TO 50
      READ (7) NX,NY,NZ,NM,K1,K2,NIT
      MX=NX+1
      MY=NY+2
      MZ=NZ+3
      IF (FDES(1).GT.0.) NM=1
      IF (FDES(NM).GT.0.) NIT=0
      DO 40 K=1,MZ
      READ (7) ((G(I,J,K),I=1,MX),J=1,MY)
   40 CONTINUE
      READ (7) (EO(K),K=K1,K2)
      REWIND 7
   50 CONTINUE
      FDIM=FHALF(1)
      NX1=NX+40-20*FDIM
      NY1=NY+2-FDIM
      NZ1=NZ+2-1*FDIM
      CALL COORD (NX,NY,NZ,KSYM,XTEO,ZTIP,XMAX,ZMAX,SY,SCAL,SCALZ,AX,AY,
```

```
     1AZ,A0,A1,A2,A3,B0,B1,B2,B3,Z,C1,C2,C3)
      CALL SINGL (NC,NZ,KSYM,KTE1,KTE2,CHORDC,SWEEP1,SWEEP2,SWEEP,DIHED1
     1,DIHED2,DIHED,ZS,XLE,YLE,XC,XZ,XZZ,YC,YZ,YZZ,Z,C1,C2,C3,E1,E2,E3,E
     24,E5,IND,CLO,CLPRE)
      CALL SURF (ND,NE,NC,NX,NZ,ISYM,KSYM,KTE1,KTE2,SCAL,YAW,A0,Z,ZS,XC,
     1YC,SLOPT,TRAIL,XS,YS,NP,ITE1,ITE2,IV,S0,Z0,XP,YP,D1,D2,D3,X,Y,IND,
     2XZ,YZ,A1,C1)
      IF (IND.EQ.0) GO TO 370
      IF (FDES(1).GT.0.) CALL SETQS (NE,NX,QPRE,S0,SOPRE,ITE1,ITE2,KTE1,
     1KTE2,Z,ZQSTA,A0,PCQ1,PCQ2,PCQ3,UC,VC,QQ1,QQ2,QQ3,QQ4,PCS1,PCS2,DSU
     2RF,NQSTA)
      IF (FCONT.GE.1.) GO TO 60
      NM=1
      NIT=0
      CALL ESTIM (ALFO)
   60 WRITE (IWRIT,670)
      FCONT=0.
      MIT=FIT(NM)+NIT
      KRES=2
      JRES=0
      NRES=0
      COV=COVO(NM)
      STRIP=STRIPO(NM)
      BETA=BETAO(NM)
      MX=NX+1
      MY=NY+2
      MZ=NZ+3
      KY=NY+1
      K1=2
      K2=NZ
      IF (KSYM.EQ.0) GO TO 70
      K1=3
      K2=NZ+2
   70 LZ=NZ/2+1
      IF (KSYM.NE.0) LZ=3
      WRITE (IWRIT,420)
      DO 80 I=2,NX
   80 WRITE (IWRIT,720) (IV(I,K),K=K1,K2)
      WRITE (IWRIT,670)
      WRITE (IWRIT,430)
      DO 90 I=2,NX
   90 WRITE (IWRIT,680) A0(I),S0(I,LZ),S0(I,KTE2)
      WRITE (IWRIT,440)
      WRITE (IWRIT,680) XMAX,AX
      WRITE (IWRIT,670)
      WRITE (IWRIT,450)
      DO 100 J=2,KY
  100 WRITE (IWRIT,680) B0(J)
      WRITE (IWRIT,460)
      WRITE (IWRIT,680) SY,AY
      WRITE (IWRIT,670)
      WRITE (IWRIT,470)
```

```
      DO 110 K=K1,K2
  110 WRITE (IWRIT,680) Z(K),XC(K),YC(K),XZ(K),YZ(K),XZZ(K),YZZ(K)
      WRITE (IWRIT,480)
      WRITE (IWRIT,680) ZMAX,AZ
      WRITE (IWRIT,670)
      WRITE (IWRIT,490)
      WRITE (IWRIT,680) STRIP
      WRITE (IWRIT,500)
      WRITE (IWRIT,710) NX,NY,NZ
      CALL SECOND (T)
      WRITE (IWRIT,770) T
      WRITE (IWRIT,510)
      WRITE (IWRIT,680) FMACH,YA,AL
      IF (FDES(NM).LE.0..AND.CLOPT.LE.0.) WRITE (IWRIT,540)
      IF (FDES(NM).GT.0.) WRITE (IWRIT,530)
      IF (CLOPT.GT.0.) WRITE (IWRIT,520)
      KDES=0
      NDES=FDES(NM)
      LX=NX/2+1
      CL=0.
      DO 120 K=K1,K2
      I1=ITE1(K)
      X(I1)=XC(K)+.5*SCAL*(AO(I1)*AO(I1)-SO(I1,K)*SO(I1,K))
      X(LX)=XC(K)+.5*SCAL*(AO(LX)*AO(LX)-SO(LX,K)*SO(LX,K))
      CHORD(K)=X(I1)-X(LX)
  120 CONTINUE
      KZDUM=KTE2-1
      S=0.
      DO 130 K=KTE1,KZDUM
      DZO=.5*(Z(K+1)-Z(K))
  130 S=S+DZO*(CHORD(K+1)+CHORD(K))
      AWING=S
  140 KDES=KDES+1
      IF (NDES.GT.0) NIT=0
  150 NIT=NIT+1
      P1=P10(NM)
      P2=P20(NM)
      P3=P30(NM)
      IF (FOPT.LT.1.) GO TO 160
      CALL OPT (QQ1,QQ2,QQ3,QQ4)
      GO TO 250
  160 CONTINUE
      CALL MIXFLO
      FCL=0.
      KCL=0
      IREFIN=0
      VOLDRG=0.
      CALL DRAGC (0,0.)
      IF (NDES.GT.0) GO TO 180
      DO 170 K=3,MZ
```

```
      IF (K.LT.KTE1.OR.K.GT.KTE2) GO TO 170
      CALL VELO (K,K,SV,SM,CP,X,Y,UC,VC,WC)
      I1=ITE1(K)
      I2=ITE2(K)
      CHORD(K)=X(I1)-X(LX)
      CALL FORCF (I1,I2,X,Y,CP,AL,CHORD(K),XC(K),SCL(K),SCD(K),SCM(K))
  170 CONTINUE
      CALL TOTFOR (KTE1,KTE2,CHORD,SCL,SCD,SCM,Z,XC,CL,CD1,CMP,CMR,CMY,A
     1WING)
  180 CONTINUE
      JO=0
      IF (NDES.LE.0) GO TO 190
      IF (NDQ.GT.0) RDQ=RDQ/FLOAT(NDQ)
      IF (CLOPT.LE.0.) WRITE (IWRIT,740) KDES,DG,IG,JG,KG,FR,IR,JR,KR,RD
     1Q,RDSO,IQ,KQ,VOLDRG,CL
  190 IF (NDES.LE.0.AND.CLOPT.LE.0.) WRITE (IWRIT,730) NIT,DG,IG,JG,KG,F
     1R,IR,JR,KR,CL,VOLDRG,P1,P2,BETA,NS
      IF (CLOPT.GT.0.) WRITE (IWRIT,750) NIT,DG,IG,JG,KG,FR,IR,JR,KR,FCL
     1,KCL,RCL,NS
      JRES=JRES+1
      IF (JRES.EQ.KRES) JRES=1
      IF (JRES.NE.1) GO TO 200
      NRES=NRES+1
      COUNT(NRES)=NIT-1
      IF (NDES.GT.0) COUNT(NRES)=MIT*KDES-1
      RES(NRES)=FR
  200 CONTINUE
      IF (NIT.LT.MIT.AND.ABS(DG).GT.COV.AND.ABS(DG).LT.10.) GO TO 150
      IF (NDES.GT.0.AND.KDES.EQ.1.AND.NIT.LT.1) GO TO 150
      IF (NDES.LE.0) GO TO 240
      RDSO=0.
      NDQ=0
      RDQ=0.
      IQ=0
      KQ=0
      DO 210 K=3,MZ
      IF (K.LT.KTE1.OR.K.GT.KTE2) GO TO 210
      CALL VELO (K,K,SV,SM,CP,X,Y,UC,VC,WC)
      I1=ITE1(K)
      I2=ITE2(K)
      CHORD(K)=X(I1)-X(LX)
      CALL FORCF (I1,I2,X,Y,CP,AL,CHORD(K),XC(K),SCL(K),SCD(K),SCM(K))
  210 CONTINUE
      CALL TOTFOR (KTE1,KTE2,CHORD,SCL,SCD,SCM,Z,XC,CL,CD1,CMP,CMR,CMY,A
     1WING)
      DO 220 I=2,NX
  220 SO(I,2)=3.*(SO(I,3)-SO(I,4))+SO(I,5)
      IF (KDES.LT.NDES) GO TO 140
      GO TO 240
  230 IF (JO.EQ.1) GO TO 10
      JO=1
      GO TO 150
```

```
240 RATE=0.
    IF (NRES.GT.1) RATE=(ABS(RES(NRES)/RES(1)))**(1./(COUNT(NRES)-COUN
   1T(1)))
    WRITE (IWRIT,550)
    WRITE (IWRIT,760) RES(1),RES(NRES),COUNT(NRES),RATE
    CALL SECOND (T)
    WRITE (IWRIT,770) T
    WRITE (IWRIT,670)
250 CONTINUE
    LX=NX/2+1
    VOLDRG=0.
    DO 260 K=K1,K2
    I1=ITE1(K)
    X(I1)=XC(K)+.5*SCAL*(AO(I1)*AO(I1)-SO(I1,K)*SO(I1,K))
    X(LX)=XC(K)+.5*SCAL*(AO(LX)*AO(LX)-SO(LX,K)*SO(LX,K))
    CHORD(K)=X(I1)-X(LX)
    SECDRG(K)=0.
    DO 260 I=2,NX
260 IDRGPLT(I,K)=-2
    IZDUM=KTE2-1
    S=0.
    DO 270 K=KTE1,IZDUM
    DZO=.5*(Z(K+1)-Z(K))
270 S=S+DZO*(CHORD(K+1)+CHORD(K))
    CALL DRAGC (0,0.)
    WRITE (IWRIT,670)
    WRITE (IWRIT,560) VOLDRG
    LX=NX/2+1
    LXO=MINO(LX+56,ITE2(KTE1))
    DO 300 I=LX,LXO
    KDUM=0
    DO 280 K=KTE1,KTE2
280 IF (IDRGPLT(I,K).EQ.-2) GO TO 290
    KDUM=KTE2
    GO TO 300
290 KDUM=K-1
300 IF (KDUM.GE.KTE1) WRITE (IWRIT,720) (IDRGPLT(I,K),K=KTE1,KDUM)
    DO 320 K=3,MZ
    IF (K.LT.KTE1.OR.K.GT.KTE2) GO TO 320
    I1=ITE1(K)
    I2=ITE2(K)
    CALL VELO (K,K,SV,SM,CP,X,Y,UC,VC,WC)
    CHORD(K)=X(I1)-X(LX)
    SECDRG(K)=SECDRG(K)/CHORD(K)
    CALL FORCF (I1,I2,X,Y,CP,AL,CHORD(K),XC(K),SCL(K),SCD(K),SCM(K))
    IF (KPLOT.GT.1.AND.K.GT.KTE1) GO TO 310
    WRITE (IWRIT,670)
    WRITE (IWRIT,570)
    WRITE (IWRIT,680) FMACH,YA,AL
    WRITE (IWRIT,580)
310 WRITE (IWRIT,680) Z(K),SCL(K),SCD(K),SECDRG(K),SCM(K),CHORD(K)
    IF (KPLOT.LE.1) CALL CPLOT (I1,I2,SM,UC,VC,QPRE(1,K),AO,SOPRE(1,K)
```

```
     1,SO(1,K),FMACH)
 320 CONTINUE
      CALL TOTFOR (KTE1,KTE2,CHORD,SCL,SCD,SCM,Z,XC,CL,CD1,CMP,CMR,CMY,A
     1WING)
      CD1=CYAW*CD1
      CD=CDO+CD1
      VLD1=0.
      IF (ABS(CD1).GT.1.E-6) VLD1=CL/CD1
      VLD=0.
      IF (ABS(CD).GT.1.E-6) VLD=CL/CD
      WRITE (IWRIT,670)
      WRITE (IWRIT,590)
      WRITE (IWRIT,680) FMACH,YA,AL
      WRITE (IWRIT,600)
      WRITE (IWRIT,680) CL,CD1,CDO,CD,VLD1,VLD
      WRITE (IWRIT,610) VOLDRG
      WRITE (IWRIT,620)
      WRITE (IWRIT,680) CMP,CMR,CMY,AWING
      IF (KPLOT.LT.1) GO TO 330
      CALL THREED (IPLOT,SV,SM,CP,X,Y,TITLE,YA,AL,VLD,CL,CD,CHORDO,XSCAL
     1,PSCAL,LABEL,NIT,UC,VC,WC,NF1)
      NF1=11
      IF (IO.EQ.0) GO TO 230
 330  IF (ISTOP.EQ.1) GO TO 380
      IF (FHALF(NM).EQ.0.) GO TO 350
      NX=NX+NX
      NY=NY+NY
      NZ=NZ+NZ
      FDIM=FHALF(2)
      NX1=NX+40-20*FDIM
      NZ1=NZ+2-1*FDIM
      NY1=NY+2-FDIM
      CALL COORD (NX,NY,NZ,KSYM,XTEO,ZTIP,XMAX,ZMAX,SY,SCAL,SCALZ,AX,AY,
     1AZ,AO,A1,A2,A3,BO,B1,B2,B3,Z,C1,C2,C3)
      CALL SINGL (NC,NZ,KSYM,KTE1,KTE2,CHORDO,SWEEP1,SWEEP2,SWEEP,DIHED1
     1,DIHED2,DIHED,ZS,XLE,YLE,XC,XZ,XZZ,YC,YZ,YZZ,Z,C1,C2,C3,E1,E2,E3,E
     24,E5,IND,CLO,CLPRE)
      CALL SURF (ND,NE,NC,NX,NZ,ISYM,KSYM,KTE1,KTE2,SCAL,YAW,AO,Z,ZS,XC,
     1YC,SLOPT,TRAIL,XS,YS,NP,ITE1,ITE2,IV,SO,ZO,XP,YP,D1,D2,D3,X,Y,IND,
     2XZ,YZ,A1,C1)
      IF (IND.EQ.0) GO TO 370
      IF (FDES(1).GT.0.) CALL SETQS (NE,NX,QPRE,SO,SOPRE,ITE1,ITE2,KTE1,
     1KTE2,Z,ZQSTA,AO,PCQ1,PCQ2,PCQ3,UC,VC,QQ1,QQ2,QQ3,QQ4,PCS1,PCS2,DSU
     2RF,NQSTA)
      CALL REFIN (ALFO)
      IREFIN=1
      IF (IO.EQ.0) GO TO 340
      N=N1
      N1=N2
      N2=N3
      N3=N
      NM=NM+1
```

```
      NIT=0
      GO TO 60
  340 NX=NX/2
      NY=NY/2
      NZ=NZ/2
      FDIM=FHALF(1)
      NX1=NX+40-20*FDIM
      NZ1=NZ+2-1*FDIM
      NY1=NY+2-FDIM
      CALL COORD (NX,NY,NZ,KSYM,XTEO,ZTIP,XMAX,ZMAX,SY,SCAL,SCALZ,AX,AY,
     1AZ,AO,A1,A2,A3,BO,B1,B2,B3,Z,C1,C2,C3)
      CALL SINGL (NC,NZ,KSYM,KTE1,KTE2,CHORDO,SWEEP1,SWEEP2,SWEEP,DIHED1
     1,DIHED2,DIHED,ZS,XLE,YLE,XC,XZ,XZZ,YC,YZ,YZZ,Z,C1,C2,C3,E1,E2,E3,E
     24,E5,IND,CLO,CLPRE)
      CALL SURF (ND,NE,NC,NX,NZ,ISYM,KSYM,KTE1,KTE2,SCAL,YAW,AO,Z,ZS,XC,
     1YC,SLOPT,TRAIL,XS,YS,NP,ITE1,ITE2,IV,SC,ZO,XP,YP,D1,D2,D3,X,Y,IND,
     2XZ,YZ,A1,C1)
      IF (IND.EQ.0) GO TO 370
      GO TO 230
  350 K1=KTE1-1
      K2=KTE2+ITE2(KTE2)-NX/2
      WRITE (8) NX,NY,NZ,NM,K1,K2,NIT
      WRITE (6,630)
      DO 360 K=1,MZ
      WRITE (8) ((G(I,J,K),I=1,MX),J=1,MY)
  360 CONTINUE
      WRITE (8) (EO(K),K=K1,K2)
      END FILE 8
      REWIND 8
      GO TO 10
  370 WRITE (IWRIT,670)
      WRITE (IWRIT,640)
      GO TO 10
  380 CONTINUE
      CALL PLOT (0.,0.,999)
      STOP 0101
C
  390 FORMAT (28HONYU INVERSE SWEPT WING CODE)
  400 FORMAT (1X,10A8)
  410 FORMAT (1X,6F10.3)
  420 FORMAT (48HOINDICATION OF LOCATION OF WING AND VORTEX SHEET,27H IN
     1 COORDINATE PLANE Y = 0./27HO((IV(I,K),K=K1,K2),I=2,NX))
  430 FORMAT (49HOCHORDWISE CELL DISTRIBUTION IN SQUARE ROOT PLANE,54H A
     1ND MAPPED SURFACE COORDINATES AT CENTER LINE AND TIP/15HO       X
     2      ,15H   ROOT PROFILE,15H   TIP PROFILE )
  440 FORMAT (15HO  TE LOCATION ,15H    POWER LAW   )
  450 FORMAT (46HONORMAL CELL DISTRIBUTION IN SQUARE ROOT PLANE/15HO
     1  Y        )
  460 FORMAT (15HO  SCALE FACTOR,15H    POWER LAW   )
  470 FORMAT (45HOSPANWISE CELL DISTRIBUTION AND SINGULAR LINE/15HO
     1  Z        ,15H      X SING    ,15H      Y SING   ,15H          XZ
     2,15H          YZ     ,15H         XZZ     ,15H          YZZ      )
```

```
480 FORMAT (15H0  TIP LOCATION,15H    POWER LAW   )
490 FORMAT (19H0ITERATIVE SOLUTION/43H0STRIP WIDTH FOR HORIZONTAL LINE
   1 RELAXATION)
500 FORMAT (15H0        NX     ,15H        NY     ,15H         NZ     )
510 FORMAT (15H0     MACH NO   ,15H        YAW    ,15H  ANG OF ATTACK)
520 FORMAT (10H0ITERATION,15H    CORRECTION ,4H  I ,4H  J ,4H  K ,15H
   1    RESIDUAL  ,4H  I ,4H  J ,4H  K ,15H  SEC LIFT COR ,4H  K ,10H
   2   RALF  ,10H SONIC PTS)
530 FORMAT (10H0ITERATION,15H    CORRECTION ,4H  I ,4H  J ,4H  K ,15H
   1    RESIDUAL  ,4H  I ,4H  J ,4H  K ,15H    AVERAGE Q  ,15H    MAXI
   2MUM Q  ,4H  I ,4H  K ,10H      DRAG,10H        CL)
540 FORMAT (10H0ITERATION,15H    CORRECTION ,4H  I ,4H  J ,4H  K ,15H
   1    RESIDUAL  ,4H  I ,4H  J ,4H  K ,10H    CL    ,10H   DRAG   ,10
   2H REL FCT 1,10H REL FCT 2,10H    BETA  ,10H SONIC PTS)
550 FORMAT (15H0 MAX RESIDAL 1,15H  MAX RESIDAL 2,15H       WORK    ,1
   15H  REDUCTN/CYCLE)
560 FORMAT (13H0WAVE DRAG = ,F9.5,/,23H PRINTOUT OF IDRAG(I,K))
570 FORMAT (24H0SECTION CHARACTERISTICS/15H0    MACH NO   ,15H        Y
   1AW     ,15H  ANG OF ATTACK)
580 FORMAT (/13H SPAN STATION,12X2HCL,10X5HCDOLD,10X5HCDNEW,13X2HCM,10
   1X5HCHORD)
590 FORMAT (21H0WING CHARACTERISTICS/15H0    MACH NO   ,15H        YAW
   1    ,15H  ANG OF ATTACK)
600 FORMAT (15H0       CL      ,15H     CD FORM   ,15H   CD FRICTION ,1
   15H       CD      ,15H     L/D FORM  ,15H       L/D      )
610 FORMAT (/2X15H     CD WAVE    ,/F10.5)
620 FORMAT (/2X8HCM PITCH,6X7HCM ROLL,9X6HCM YAW,9X5HAWING)
630 FORMAT (1X,14HWRITE ON TAPE8)
640 FORMAT (24H0BAD DATA,SPLINE FAILURE)
650 FORMAT (8E10.7)
660 FORMAT (10A8)
670 FORMAT (1H1)
680 FORMAT (F12.5,7F15.5)
690 FORMAT (1X,8E15.5)
700 FORMAT (1H0,10A8)
710 FORMAT (I8,7I15)
720 FORMAT (1X,32I4)
730 FORMAT (I10,E15.5,3I4,E15.5,3I4,5F10.5,I10)
740 FORMAT (I10,E15.5,3I4,E15.5,3I4,2E15.5,2I4,F10.4,F6.2)
750 FORMAT (I10,E15.5,3I4,E15.5,3I4,E15.5,I4,F10.5,I10)
760 FORMAT (2E15.4,2F15.4)
770 FORMAT (15H0COMPUTING TIME,F10.3,10H   SECONDS)
780 FORMAT (/5X3HFNX,11X3HFNY,11X3HFNZ,10X5HFPLOT,9X5HXSCAL,9X5HPSCAL,
   19X5HFCONT,10X6HFSWEEP/1X,8E14.5)
790 FORMAT (/4X7HFIT(NM),8X8HCOVO(NM),8X7HP10(NM),8X7HP20(NM),8X7HP30(
   1NM),6X9HBETAO(NM),6X9HFHALF(NM),6X8HFDES(NM))
800 FORMAT (/5X5HFMACH,11X2HYA,14X2HAL,12X3HCDO,12X4HSREF/1X,5E15.5)
    END
```

```
      SUBROUTINE GEOM (ND,NC,NP,ZS,XS,YS,XLE,YLE,SLOPT,TRAIL,XP,YP,SWEEP
     11,SWEEP2,SWEEP,DIHED1,DIHED2,DIHED,XTEO,CHORDO,ZTIP,ISYMO,KSYM,CLO
     2)
C     GEOMETRIC DEFINITION OF WING
      DIMENSION XS(ND,1), YS(ND,1), ZS(1), XLE(1), YLE(1), SLOPT(1), TRA
     1IL(1), XP(1), YP(1), NP(1), CLO(1)
      DIMENSION DESC(10)
      IREAD=5
      IWRIT=6
      RAD=57.2957795130823
      READ (IREAD,150) DESC
      READ (IREAD,140) ZSYM,FNC,SWEEP1,SWEEP2,SWEEP,DIHED1,DIHED2,DIHED
      WRITE (IWRIT,190) ZSYM,FNC,SWEEP1,SWEEP2,SWEEP,DIHED1,DIHED2,DIHED
      IF (FNC.LT.3.) RETURN
      KSYM=ZSYM
      NC=FNC
      SWEEP1=SWEEP1/RAD
      SWEEP2=SWEEP2/RAD
      SWEEP=SWEEP/RAD
      DIHED1=DIHED1/RAD
      DIHED2=DIHED2/RAD
      DIHED=DIHED/RAD
      ISYMO=1
      XTEO=0.
      CHORDO=0.
      K=1
   10 READ (IREAD,150) DESC
      READ (IREAD,140) ZS(K),XL,YL,CHORD,THICK,AL,FSEC,CLO(K)
      WRITE (IWRIT,200) ZS(K),XL,YL,CHORD,THICK,AL,FSEC
      ALPHA=AL/RAD
      IF (K.GT.1.AND.FSEC.EQ.0.) GO TO 80
      READ (IREAD,150) DESC
      READ (IREAD,140) YSYM,FNU,FNL,SNGOPT,ZEND,SNGRAT
      WRITE (IWRIT,210) YSYM,FNU,FNL
      NU=FNU
      NL=FNL
      N=NU+NL-1
      READ (IREAD,150) DESC
      READ (IREAD,140) TRL,SLT,XSING,YSING
      WRITE (IWRIT,220) TRL,SLT,XSING,YSING
      READ (IREAD,150) DESC
      WRITE (IWRIT,230)
      DO 20 I=NL,N
      READ (IREAD,140) XP(I),YP(I)
   20 WRITE (IWRIT,180) XP(I),YP(I)
      L=NL+1
      IF (YSYM.GT.0.) GO TO 40
```

```
      READ (IREAD,150) DESC
      WRITE (IWRIT,240)
      DO 30 I=1,NL
      READ (IREAD,140) VAL,DUM
      WRITE (IWRIT,180) VAL,DUM
      J=L-I
      XP(J)=VAL
   30 YP(J)=DUM
      GO TO 60
   40 J=L
      DO 50 I=NL,N
      J=J-1
      XP(J)=XP(I)
   50 YP(J)=-YP(I)
   60 WRITE (IWRIT,160)
      WRITE (IWRIT,110) ZS(K)
      WRITE (IWRIT,170) TRL,SLT,XSING,YSING
      WRITE (IWRIT,120)
      DO 70 I=1,N
   70 WRITE (IWRIT,170) XP(I),YP(I)
   80 CONTINUE
      SCALE=CHORD/(XP(1)-XP(NL))
      XLE(K)=XL+(XSING-XP(NL))*THICK*SCALE
      YLE(K)=YL+(YSING-YP(NL))*THICK*SCALE
      XX=XP(NL)+(XSING-XP(NL))*THICK
      YY=YP(NL)+(YSING-YP(NL))*THICK
      CA=COS(ALPHA)
      SA=SIN(ALPHA)
      DO 90 I=1,N
      XS(I,K)=SCALE*((XP(I)-XX)*CA+THICK*(YP(I)-YY)*SA)
   90 YS(I,K)=SCALE*(THICK*(YP(I)-YY)*CA-(XP(I)-XX)*SA)
      SLOPT(K)=THICK*SLT-TAN(ALPHA)
      TRAIL(K)=THICK*TRL/RAD
      NP(K)=N
      XTEO=AMAX1(XTEO,XS(1,K))
      CHORDO=AMAX1(CHORDO,CHORD)
      IF (YSYM.LE.0..OR.ALPHA.NE.0.) ISYMO=0
      WRITE (IWRIT,130) ZS(K)
      WRITE (IWRIT,170) XL,YL,CHORD,THICK,AL
      K=K+1
      IF (K.LE.NC) GO TO 10
      ZO=.5*(ZS(1)+ZS(NC))
      IF (KSYM.NE.0) ZO=ZS(1)
      DO 100 K=1,NC
  100 ZS(K)=ZS(K)-ZO
      ZTIP=ZS(NC)
      RETURN
C
  110 FORMAT (16HOPROFILE AT Z = ,F10.5/15HO    TE ANGLE   ,15H      TE SL
     1OPE   ,15H       X SING    ,15H       Y SING    )
  120 FORMAT (15HO       X       ,15H        Y        )
  130 FORMAT (27HOSECTION DEFINITION AT Z = ,F10.5/15HO      XLE      ,15
```

```
   1H         YLE       ,15H       CHCRD     ,15HTHICKNESS RATIO,15H       AL
   2PHA      )
140 FORMAT (8F10.6)
150 FORMAT (10A8)
160 FORMAT (1H1)
170 FORMAT (F12.4,7F15.4)
180 FORMAT (8E15.5)
190 FORMAT (/5X4HZSYM,12X3HFNC,10X6HSWEEP1,9X6HSWEEP2,9X5HSWEEP,10X6HD
   1IHED1,9X6HDIHED2,10X5HDIHED/1X,8E15.5)
200 FORMAT (/5X5HZS(K),12X2HXL,13X2HYL,11X5HCHORD,10X5HTHICK,12X2HAL,1
   12X4HFSEC/1X,7E15.5)
210 FORMAT (/6X4HYSYM,11X3HFNU,12X3HFNL/1X,3E15.5)
220 FORMAT (/6X3HTRL,12X3HSLT,11X5HXSING,10X5HYSING/1X,4E15.5)
230 FORMAT (/5X5HXP(I),10X5HYP(I))
240 FORMAT (/6X3HVAL,12X3HDUM)
    END
```

```
    SUBROUTINE COORD (NX,NY,NZ,KSYM,XTEO,ZTIP,XMAX,ZMAX,SY,SCAL,SCALZ,
   1AX,AY,AZ,A0,A1,A2,A3,B0,B1,B2,B3,Z,C1,C2,C3)
C   SETS UP STRETCHED PARABOLIC AND SPANWISE COORDINATES
    DIMENSION A0(1), A1(1), A2(1), A3(1), B0(1), B1(1), B2(1), B3(1),
   1Z(1), C1(1), C2(1), C3(1)
    COMMON /DIM/ NX1,NY1,NZ1,FDIM
    DX=2./NX
    DY=1./NY
    KY=NY+1
    DZ=2./NZ
    DX=2./NX1
    DY=1./NY1
    DZ=2./NZ1
    KY1=NY1+1
    Z0=1.-DZ
    K1=2
    K2=NZ
    IF (KSYM.EQ.0) GO TO 10
    DZ=1./NZ
    DZ=1./NZ1
    Z0=0.
    K1=3
    K2=NZ+2
 10 AX=.5
    AY=.5
    AZ=.5
    BX=0.
    BZ=0.
```

```
      XMAX=.625
      ZMAX=8./14.
      SY=.5
      SCAL=XTEO/(.50001*XMAX*XMAX)
      SCALZ=ZTIP/(1.000001*ZMAX)
      V2=(DX/DY)**2
      W1=SCAL/SCALZ
      W2=(W1*DX/DZ)**2
      S73=SQRT(73.)
      BBX=-BX*SQRT(3.*(7.+S73))/((1.+S73)*XMAX**3)
      ABX=1.-BBX*SQRT((7.+S73)/12.)*XMAX**3
      CBX=(19.+S73)*XMAX*XMAX/12.
      ABBX=ABX+BBX*(3.*CBX-4.*XMAX*XMAX)*XMAX*XMAX/SQRT(CBX-XMAX*XMAX)
      MX=NX+1
      DO 40 I=1,MX
      DD=(I-1)*DX-1.+(NX1-NX)*DX/2.
      B=1.
      IF (ABS(DD).GT.XMAX) GO TO 20
      A=CBX-DD*DD
      AS=SQRT(A)
      C=ABX*AS+BBX*(3.*CBX-4.*DD*DD)*DD*DD
      DO=ABX*DD+BBX*AS*DD**3
      D1=AS/C
      D2=BBX*(CBX*(-6.*CBX+19.*DD*DD)-12.*DD**4)*DD/(A*C)
      GO TO 30
   20 IF (DD.LT.0.) B=-1.
      A=1.-((DD-B*XMAX)/(1.-XMAX))**2
      C=A**AX
      D=(AX+AX-1.)*(1.-A)
      DO=B*XMAX+ABBX*(DD-B*XMAX)/C
      D1=A*C/((1.+D)*ABBX)
      D2=-(AX+AX)*(DD-B*XMAX)*(3.+D)/((1.+D)*A*(1.-XMAX)**2)
   30 AO(I)=DO
      A1(I)=.5*D1/DX
      A2(I)=D1*D1
      A3(I)=.5*DX*D2
   40 CONTINUE
      JS=4-FDIM
      DO 50 JJ=JS,KY1
      J=JJ-(2-FDIM)
      DD=(KY1-JJ)*DY
      A=1.-DD*DD
      C=A**AY
      D=(AY+AY-1.)*(1.-A)
      D1=A*C/((1.+D)*SY)
      BO(J)=SY*DD/C
      B1(J)=.5*D1/DY
      B2(J)=D1*D1*V2
   50 B3(J)=-AY*DD*DY*(3.+D)/((1.+D)*A)
      BBZ=-BZ*SQRT(3.*(7.+S73))/((1.+S73)*ZMAX**3)
      ABZ=1.-BBZ*SQRT((7.+S73)/12.)*ZMAX**3
      CBZ=(19.+S73)*ZMAX*ZMAX/12.
```

```
      ABBZ=ABZ+BBZ*(3.*CBZ-4.*ZMAX*ZMAX)*ZMAX*ZMAX/SQRT(CBZ-ZMAX*ZMAX)
      DO 80 K=2,K2
      DD=(K-K1)*DZ-ZO
      B=1.
      IF (ABS(DD).GT.ZMAX) GO TO 60
      A=CBZ-DD*DD
      AS=SQRT(A)
      C=ABZ*AS+BBZ*(3.*CBZ-4.*DD*DD)*DD*DD
      DO=ABZ*DD+BBZ*AS*DD**3
      D1=AS/C
      D2=BBZ*(CBZ*(-6.*CBZ+19.*DD*DD)-12.*DD**4)*DD/(A*C)
      GO TO 70
   60 IF (DD.LT.O.) B=-1.
      A=1.-((DD-B*ZMAX)/(1.-ZMAX))**2
      C=A**AZ
      D=(AZ+AZ-1.)*(1.-A)
      DO=B*ZMAX+ABBZ*(DD-B*ZMAX)/C
      D1=A*C/((1.+D)*ABBZ)
      D2=-(AZ+AZ)*(DD-B*ZMAX)*(3.+D)/((1.+D)*A*(1.-ZMAX)**2)
   70 Z(K)=SCALZ*DO
      C1(K)=.5*D1*W1/DZ
      C2(K)=D1*D1*W2
      C3(K)=.5*DZ*D2
   80 CONTINUE
      RETURN
      END



      SUBROUTINE SINGL (NC,NZ,KSYM,KTE1,KTE2,CHORDO,SWEEP1,SWEEP2,SWEEP,
     1DIHED1,DIHED2,DIHED,ZS,XLE,YLE,XC,XZ,XZZ,YC,YZ,YZZ,Z,C1,C2,C3,E1,E
     22,E3,E4,E5,IND,CLO,CLPRE)
C     GENERATES SINGULAR LINE FOR SQUARE ROOT TRANSFORMATION
      DIMENSION ZS(1), XLE(1), YLE(1), XC(1), XZ(1), XZZ(1), YC(1), YZ(1
     1), YZZ(1), Z(1), C1(1), C2(1), C3(1), E1(1), E2(1), E3(1), E4(1),
     2E5(1), CLO(1), CLPRE(1)
      DO 10 K=1,NC
      E4(K)=0.
   10 E5(K)=0.
      K1=2
      K2=NZ
      IF (KSYM.EQ.O) GO TO 20
      K1=3
      K2=NZ+2
      KTE1=3
   20 DO 30 K=K1,K2
      IF (Z(K).LT.ZS(1)) KTE1=K+1
```

```
      IF (Z(K).LE.ZS(NC)) KTE2=K
   30 CONTINUE
      B=CHORDO
      S1=TAN(SWEEP1)
      S2=TAN(SWEEP2)
      T1=TAN(DIHED1)
      T2=TAN(DIHED2)
      CALL SPLIF (1,NC,ZS,XLE,E1,E2,E3,1,S1,1,S2,0,0.,IND)
      CALL INTPL (KTE1,KTE2,Z,XC,1,NC,ZS,XLE,E1,E2,E3,0)
      CALL INTPL (KTE1,KTE2,Z,XZ,1,NC,ZS,E1,E2,E3,E4,0)
      CALL INTPL (KTE1,KTE2,Z,XZZ,1,NC,ZS,E2,E3,E4,E5,0)
      CALL SPLIF (1,NC,ZS,YLE,E1,E2,E3,1,T1,1,T2,0,0.,IND)
      CALL INTPL (KTE1,KTE2,Z,YC,1,NC,ZS,YLE,E1,E2,E3,0)
      CALL INTPL (KTE1,KTE2,Z,YZ,1,NC,ZS,E1,E2,E3,E4,0)
      CALL INTPL (KTE1,KTE2,Z,YZZ,1,NC,ZS,E2,E3,E4,E5,0)
      CALL SPLIF (1,NC,ZS,CLO,E1,E2,E3,3,0.,3,0.,0,0.,IND)
      CALL INTPL (KTE1,KTE2,Z,CLPRE,1,NC,ZS,CLO,E1,E2,E3,0)
      S=B*TAN(SWEEP)
      S1=B*S1
      S2=B*S2
      T=B*TAN(DIHED)
      T1=B*T1
      T2=B*T2
      XC(2)=3.*(XC(3)-XC(4))+XC(5)
      YC(2)=3.*(YC(3)-YC(4))+YC(5)
      IF (KSYM.NE.0) GO TO 50
      N=KTE1-1
      DO 40 K=K1,N
      ZZ=(Z(K)-Z(KTE1))/B
      A=EXP(ZZ)
      XC(K)=XC(KTE1)+S*ZZ-(S1-S)*(1.-A)
      YC(K)=YC(KTE1)+T*ZZ-(T1-T)*(1.-A)
      XZ(K)=(S+(S1-S)*A)/B
      YZ(K)=(T+(T1-T)*A)/B
      XZZ(K)=(S1-S)*A/(B*B)
   40 YZZ(K)=(T1-T)*A/(B*B)
   50 N=KTE2+1
      DO 60 K=N,K2
      ZZ=(Z(K)-Z(KTE2))/B
      A=EXP(-ZZ)
      XC(K)=XC(KTE2)+S*ZZ+(S2-S)*(1.-A)
      YC(K)=YC(KTE2)+T*ZZ+(T2-T)*(1.-A)
      XZ(K)=(S+(S2-S)*A)/B
      YZ(K)=(T+(T2-T)*A)/B
      XZZ(K)=-(S2-S)*A/(B*B)
   60 YZZ(K)=-(T2-T)*A/(B*B)
      RETURN
      END
```

```
      SUBROUTINE SURF (ND,NE,NC,NX,NZ,ISYM,KSYM,KTE1,KTE2,SCAL,YAW,AO,Z,
     1ZS,XC,YC,SLOPT,TRAIL,XS,YS,NP,ITE1,ITE2,IV,SO,ZO,XP,YP,D1,D2,D3,X,
     2Y,IND,XZ,YZ,A1,C1)
C     INTERPOLATES MAPPED WING SURFACE AT MESH POINTS
C     INTERPOLATION IS LINEAR IN PHYSICAL PLANE
      DIMENSION SO(NE,1), XS(ND,1), YS(ND,1), ZS(1), SLOPT(1), TRAIL(1),
     1 XC(1), YC(1), AO(1), Z(1), ZO(1), X(1), Y(1), XP(1), YP(1), D1(1)
     2, D2(1), D3(1), IV(NE,1), NP(1), ITE1(1), ITE2(1), XZ(1), YZ(1), A
     31(1), C1(1)
      COMMON /DIM/ NX1,NY1,NZ1,FDIM
      PI=3.14159265268979
      TYAW=TAN(YAW)
      S1=.5*SCAL
      DX=2./NX1
      LX=NX/2+1
      MX=NX+1
      MZ=NZ+3
      IVO=1-ISYM-ISYM-ISYM
      IV1=-1-ISYM
      DO 10 K=1,MZ
      ITE1(K)=MX
      ITE2(K)=MX
      DO 10 I=1,MX
      IV(I,K)=-2
   10 SO(I,K)=0.
      K=KTE1
      K2=1
   20 K2=K2+1
      K1=K2-1
      R2=1.
      IF (ZS(K2)-Z(K)) 20,40,30
   30 R2=(Z(K)-ZS(K1))/(ZS(K2)-ZS(K1))
   40 R1=1.-R2
      C=R1*XS(1,K1)+R2*XS(1,K2)
      CC=SQRT((C+C)/SCAL)
      DO 50 I=2,NX
      IF ((AO(I)+.5*DX).LT.-CC) I1=I+1
      IF ((AO(I)-.5*DX).LT.CC) I2=1
   50 CONTINUE
      ITE1(K)=I1
      ITE2(K)=I2
      CC=AO(I2)/CC
      ZO(K)=Z(K)-TYAW*(XC(K)+S1*AO(I2)*AO(I2))
      KK=K1
      P=R1
   60 N=NP(KK)
      Q=SQRT(XS(1,KK)/C)/CC
```

```
      DO 70 I=2,NX
   70 X(I)=Q*AO(I)
      ANGL=PI+PI
      U=1.
      V=0.
      DO 90 I=1,N
      R=SQRT(XS(I,KK)**2+YS(I,KK)**2)
      IF (R.EQ.0.) GO TO 80
      ANGL=ANGL+ATAN2((U*YS(I,KK)-V*XS(I,KK)),(U*XS(I,KK)+V*YS(I,KK)))
      U=XS(I,KK)
      V=YS(I,KK)
      R=SQRT((R+R)/SCAL)
      XP(I)=R*COS(.5*ANGL)
      YP(I)=R*SIN(.5*ANGL)
      GO TO 90
   80 ANGL=PI
      U=-1.
      V=0.
      XP(I)=0.
      YP(I)=0.
   90 CONTINUE
      ANGL=ATAN(SLOPT(KK))
      ANGL1=ATAN(YS(1,KK)/XS(1,KK))
      ANGL2=ATAN(YS(N,KK)/XS(N,KK))
      ANGL1=ANGL-.5*(ANGL1-TRAIL(KK))
      ANGL2=ANGL-.5*(ANGL2+TRAIL(KK))
      T1=TAN(ANGL1)
      T2=TAN(ANGL2)
      CALL SPLIF (1,N,XP,YP,D1,D2,D3,1,T1,1,T2,0,0.,IND)
      CALL INTPL (I1,I2,X,Y,1,N,XP,YP,D1,D2,D3,0)
      X1=.25*XS(1,KK)
      A=SLOPT(KK)*(XS(1,KK)-X1)
      B=1./(XS(1,KK)-X1)
      ANGL=PI+PI
      U=1.
      V=0.
      M=I1-1
      DO 100 I=2,M
      XX=.5*SCAL*X(I)**2
      D=B*(XX-X1)
      YY=YS(1,KK)+A*ALOG(D)/D
      R=SQRT(XX**2+YY**2)
      ANGL=ANGL+ATAN2((U*YY-V*XX),(U*XX+V*YY))
      U=XX
      V=YY
      R=SQRT((R+R)/SCAL)
  100 Y(I)=R*SIN(.5*ANGL)
      A=SLOPT(KK)*(XS(N,KK)-X1)
      B=1./(XS(N,KK)-X1)
      ANGL=0.
      U=1.
      V=0.
```

```
      M=I2+1
      DO 110 I=M,NX
      XX=.5*SCAL*X(I)**2
      D=B*(XX-X1)
      YY=YS(N,KK)+A*ALOG(D)/D
      R=SQRT(XX**2+YY**2)
      ANGL=ANGL+ATAN2((U*YY-V*XX),(U*XX+V*YY))
      U=XX
      V=YY
      R=SQRT((R+R)/SCAL)
  110 Y(I)=R*SIN(.5*ANGL)
      Q=P*Q*CC*CC
      DO 120 I=2,NX
  120 SO(I,K)=SO(I,K)+Q*Y(I)
      IF (KK.EQ.K2) GO TO 130
      KK=K2
      P=R2
      GO TO 60
  130 DO 140 I=I1,I2
  140 IV(I,K)=2
      M=I1-1
      DO 150 I=2,M
      ZZ=Z(K)-TYAW*(XC(K)+S1*AO(I)*AO(I))
      IF (ZZ.GE.ZO(KTE1)) IV(I,K)=IVO
  150 CONTINUE
      M=I2+1
      DO 160 I=M,NX
      ZZ=Z(K)-TYAW*(XC(K)+S1*AO(I)*AO(I))
      IF (ZZ.GE.ZO(KTE1)) IV(I,K)=IVO
  160 CONTINUE
      K2=K2-1
      K=K+1
      IF (K.LE.KTE2) GO TO 20
      K1=2
      K2=NZ
      IF (KSYM.EQ.0) GO TO 170
      K1=3
      K2=NZ+2
  170 DO 180 I=2,NX
      ZZ=Z(K)-TYAW*(XC(K)+S1*AO(I)*AO(I))
      IF (ZZ.LE.ZS(NC).AND.ZZ.GE.ZO(KTE1)) IV(I,K)=IVO
  180 CONTINUE
      K=K+1
      IF (K.LE.K2) GO TO 170
      N=KTE2
      IF (YAW.LE.0.) GO TO 200
      IO=ITE1(KTE2)+1
      DO 190 I=IO,LX
      N=N+1
  190 ZO(N)=Z(KTE2)-TYAW*(XC(KTE2)+S1*AO(I)*AO(I))
  200 I=ITE1(KTE1)
      ZO(KTE1-1)=Z(KTE1-1)-TYAW*(XC(KTE1-1)+S1*AO(I)*AO(I))
```

```
      ZO(N+1)=Z(KTE2+1)
      DO 220 K=K1,K2
      DO 210 I=2,NX
      IF (IV(I,K).GT.0) GO TO 210
      IF (IV(I+1,K+1).GT.0.OR.IV(I-1,K+1).GT.0) IV(I,K)=IV1
      IF (IV(I+1,K-1).GT.0.OR.IV(I-1,K-1).GT.0) IV(I,K)=IV1
  210 CONTINUE
  220 IF (SO(LX,K).LT.1.E-05) IV(LX,K)=0
      IF (KSYM.EQ.0) RETURN
      DO 230 I=2,NX
  230 SO(I,2)=3.*(SO(I,3)-SO(I,4))+SO(I,5)
      RETURN
      END



      SUBROUTINE ESTIM (ALFO)
C     INITIAL ESTIMATE OF REDUCED POTENTIAL
      COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
     1,ITE2(15),A0(129),A1(129),A2(129),A3(129),B0(12),B1(12),B2(12),B3(
     212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
     3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
     4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
     515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
     6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,F00,F01,F10,F11,
     7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
      DIMENSION ALFO(1)
      KY=NY+1
      MZ=NZ+3
      DO 10 I=1,129
      DO 10 J=1,12
      DO 10 K=1,15
   10 G(I,J,K)=0.
      K=1
      DO 30 K=1,MZ
      DO 20 I=2,NX
      G(I,KY+1,K)=0.
      IF (IV(I,K).LT.2) GO TO 20
      DSI=SO(I+1,K)-SO(I-1,K)
      DSK=SO(I,K+1)-SO(I,K-1)
      SX=A1(I)*DSI
      SZ=C1(K)*DSK
      FH=A0(I)*A0(I)+SO(I,K)*SO(I,K)
      H=1./FH
      AZ=-A0(I)*XZ(K)-SO(I,K)*YZ(K)
      BZ=-A0(I)*YZ(K)+SO(I,K)*XZ(K)
      HZ=AZ*SX-BZ+FH*SZ
```

```
      FYY=1.+SX*SX+H*HZ*HZ
      FXY=SX+H*AZ*HZ
      V=SA*AO(I)-CA*SO(I,K)
      U=CA*AO(I)+SA*SO(I,K)
      W=SYAW+CA*XZ(K)+SA*YZ(K)
      G(I,KY+1,K)=G(I,KY-1,K)+(V*(1.-H*BZ*HZ)-U*FXY-W*HZ)/(FYY*B1(KY))
   20 CONTINUE
   30 CONTINUE
      K1=KTE1-1
      K2=KTE2+ITE2(KTE2)-NX/2
      DO 40 K=K1,K2
      ALFO(K)=0.
   40 EO(K)=0.
      IO=1
      RETURN
      END


      SUBROUTINE MIXFLO
C     SOLUTION OF EQUATIONS FOR MIXED SUBSONIC AND SUPERSONIC FLOW
C     USING ROTATED DIFFERENCE SCHEME
      COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
     1,ITE2(15),AO(129),A1(129),A2(129),A3(129),BO(12),B1(12),B2(12),B3(
     212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
     3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
     4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
     515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
     6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,FOO,FO1,F10,F11,
     7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
      COMMON /FLO/ STRIP,P1,P2,P3,BETA,FR,IR,JR,KR,DG,IG,JG,KG,NS,FSWEEP
      COMMON /SWP/ DXYZ(129),GK1(129,15),GK2(129,15),SX(129),SZ(129),SXX
     1(129),SXZ(129),SZZ(129),RO(129),R1(129),C(129),D(129),G10(15),G20(
     215),G30(15),G40(15),G1(15),G2(15),I1,I2,K,L,NO,LX,MX,KY,MY,T1,AAO,
     3Q1,Q2,TYAW,S1
      COMMON /DIM/ NX1,NY1,NZ1,FDIM
      BETX=.01
      BETY=.15
      BETZ=.1
      BSCAL=1./(1.+FDIM)
      BSCAL1=1./(2.*(1.+FDIM))
      LX=NX/2+1
      MX=NX+1
      KY=NY+1
```

```
      MY=NY+2
      TYAW=SYAW/CYAW
      S1=.5*SCAL
      DX=2./NX1
      T1=DX*DX
      AA0=1./FMACH**2+.2
      Q1=2./P1
      Q2=1./P2
      FR=0.
      IR=0
      JR=0
      KR=0
      DG=0.
      IG=0
      JG=0
      KG=0
      NS=0
      K1=2
      IF (FMACH.GE.1.) K1=3
      K2=NZ
      IF (KSYM.EQ.0) GO TO 10
      K1=3
      K2=NZ+2
   10 F=ABS(.5*STRIP*NX)
      L=F
      IF (L.EQ.NX/2) L=L-1
      I1=LX-L
      I2=LX+L
      IF (L.EQ.0) I2=LX-1
      DO 20 J=1,MY
      DO 20 I=1,MX
      GK1(I,J)=G(I,J,1)
   20 GK2(I,J)=G(I,J,1)
      K=2
      L=2
      NO=KTE1-1
      IF (K.EQ.K1) GO TO 90
      IF (KSYM.EQ.0) GO TO 80
      I=LX
      DSI=SO(I+1,3)-SO(I-1,3)
      DSK=SO(I,4)-SO(I,2)
      SX(I)=A1(I)*DSI
      SZ(I)=C1(3)*DSK
      R=1.0
      DO 30 J=2,KY
      YP=BO(J)+SO(I,3)
      IF (J.EQ.KY) R=AMINO(1,IV(I,K))
      H=R/(1.-R+YP*YP)
      AZ=-YP*YZ(3)
      BZ=YP*XZ(3)
      A=H*AZ*A1(I)
      B=(H*(BZ-AZ*SX(I))-SZ(I))*B1(J)
```

```
      DGI=G(I+1,J,3)-G(I-1,J,3)
      DGJ=G(I,J+1,3)-G(I,J-1,3)
      G(I,J,2)=G(I,J,4)+(A*DGI-B*DGJ)/C1(3)
      GK1(I,J)=G(I,J,2)
      G(I,J,1)=3.*(G(I,J,2)-G(I,J,3))+G(I,J,4)
      GK2(I,J)=G(I,J,1)
   30 CONTINUE
      J=KY+1
      G(I,J,2)=G(I,J,4)+(A*DGI-B*DGJ)/C1(3)
      GK1(I,J)=G(I,J,2)
      G(I,J,1)=3.*(G(I,J,2)-G(I,J,3))+G(I,J,4)
      GK2(I,J)=G(I,J,1)
      M=NX/2-1
      DO 70 II=1,M
      I=LX-II
      GO TO 50
   40 I=LX+II
   50 DSI=SO(I+1,3)-SO(I-1,3)
      DSK=SO(I,4)-SO(I,2)
      SX(I)=A1(I)*DSI
      SZ(I)=C1(3)*DSK
      DO 60 J=2,KY
      YP=BO(J)+SO(I,3)
      H=1./(AO(I)*AO(I)+YP*YP)
      AZ=-AO(I)*XZ(3)-YP*YZ(3)
      BZ=-AO(I)*YZ(3)+YP*XZ(3)
      S=SIGN(1.,AZ)
      A=H*ABS(AZ)*A1(I)
      B=(H*(BZ-AZ*SX(I))-SZ(I))*B1(J)
      IP=I+IFIX(S)
      IM=I-IFIX(S)
      DGI=G(I,J,4)-G(IM,J,4)
      DGJ=G(I,J+1,3)-G(I,J-1,3)
      G(I,J,2)=(C1(3)*G(I,J,4)+A*(G(IP,J,2)+DGI)-B*DGJ)/(C1(3)+A)
      GK1(I,J)=G(I,J,2)
      G(I,J,1)=3.*(G(I,J,2)-G(I,J,3))+G(I,J,4)
   60 GK2(I,J)=G(I,J,1)
      J=KY+1
      G(I,J,2)=(C1(3)*G(I,J,4)+A*(G(IP,J,2)+DGI)-B*DGJ)/(C1(3)+A)
      GK1(I,J)=G(I,J,2)
      IF (I.LT.LX) GO TO 40
   70 CONTINUE
   80 KK=K+1
      K3=K2+1
   90 DO 150 K=KK,K2
      DO 100 J=1,MY
      G10(J)=G(I2,J,K)
      G20(J)=G(I2-1,J,K)
      G30(J)=G(I1,J,K)
  100 G40(J)=G(I1+1,J,K)
      DO 110 I=2,NX
      DSI=SO(I+1,K)-SO(I-1,K)
```

```
      DSK=SO(I,K+1)-SO(I,K-1)
      DSII=SO(I+1,K)-SO(I,K)-SO(I,K)+SO(I-1,K)+A3(I)*DSI
      DSKK=SO(I,K+1)-SO(I,K)-SO(I,K)+SO(I,K-1)+C3(K)*DSK
      DSIK=SO(I+1,K+1)-SO(I-1,K+1)-SO(I+1,K-1)+SO(I-1,K-1)
      SX(I)=A1(I)*DSI
      SZ(I)=C1(K)*DSK
      SXX(I)=A2(I)*DSII
      SZZ(I)=C2(K)*DSKK
  110 SXZ(I)=T1*A1(I)*C1(K)*DSIK
      L=K
      IF (I2.LE.I1) GO TO 130
      IF (FSWEEP.LT.0.) GO TO 120
      CALL YSWEEP
      GO TO 130
  120 CALL VYSWEEP
  130 CONTINUE
      IF (K.NE.KTE2.OR.YAW.LE.0.) GO TO 150
      I0=ITE1(K)+1
      DO 140 I=I0,LX
      M=NX+2-I
      E=G(M,KY,K)-G(I,KY,K)
      NO=NO+1
  140 EO(NO)=EO(NO)+P3*(E-EO(NO))
  150 CONTINUE
C     BOUNDARY CONDITION AT INFINITY REPLACED BY MIXED DIRICHLET
C     AND NEUMANN CONDITION AT CONTROL SURFACE
      DO 160 I=1,MX
      DO 160 J=1,MY
  160 G(I,J,K3)=(1.-BETZ/BSCAL1)*G(I,J,K2)
      DO 170 J=1,MY
      G(I2+1,J,2)=(1.-BETX/BSCAL)*G(I2,J,2)
  170 G(I1-1,J,2)=(1.-BETX/BSCAL)*G(I1,J,2)
      DO 180 I=1,MX
  180 G(I,J1-1,2)=(1.-BETY/BSCAL1)*G(I,J1,2)
      FR=1.2*FR/AAO
      RETURN
      END


      SUBROUTINE YSWEEP
C     THE FINITE DIFFERENCE EQUATIONS FOR G ARE SOLVED BY ROW RELAXATION
C     MOST OF THE COMPUTING TIME IS SPENT IN THIS ROUTINE
      COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
     1,ITE2(15),AO(129),A1(129),A2(129),A3(129),BO(12),B1(12),B2(12),B3(
     212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
     3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
```

```
4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,F00,F01,F10,F11,
7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
 COMMON /FLO/ STRIP,P1,P2,P3,BETA,FR,IR,JR,KR,DG,IG,JG,KG,NS,FSWEEP
 COMMON /SWP/ DXYZ(129),GK1(129,15),GK2(129,15),SX(129),SZ(129),SXX
1(129),SXZ(129),SZZ(129),R0(129),R1(129),C(129),D(129),G10(15),G20(
215),G30(15),G40(15),G1(15),G2(15),I1,I2,K,L,NO,LX,MX,KY,MY,T1,AAO,
3Q1,Q2,TYAW,S1
 COMMON /DIM/ NX1,NY1,NZ1,FDIM
 BETX=.01
 BETY=.15
 BETZ=.1
 BSCAL=1./(1.+FDIM)
 BSCAL1=1./(2.*(1.+FDIM))
 J1=2
 IF (FMACH.GE.1.) J1=3
 C(I1-1)=0.
 D(I1-1)=0.
 DO 10 I=I1,I2
 R0(I)=1.
 R1(I)=1.
 GK1(I,1)=G(I,1,L)
10 GK1(I,J1-1)=G(I,J1-1,L)
 J=J1
 I3=I2
20 BC=-T1*B1(J)*C1(K)
 DO 60 I=I1,I3
 AB=-T1*A1(I)*B1(J)
 AC=T1*A1(I)*C1(K)
 YP=S0(I,K)+B0(J)
 A=1.-R0(I)+A0(I)*A0(I)+YP*YP
 H=R0(I)/A
 FH=R0(I)*A
 P=A0(I)*(4.*YP*YP-FH)
 Q=YP*(4.*A0(I)*A0(I)-FH)
 A=XZ(K)*XZ(K)-YZ(K)*YZ(K)
 B=(XZ(K)+XZ(K))*YZ(K)
 AZ=-A0(I)*XZ(K)-YP*YZ(K)
 BZ=-A0(I)*YZ(K)+YP*XZ(K)
 CZ=H*H*(P*A-Q*B)-A0(I)*XZZ(K)-YP*YZZ(K)
 DZ=H*H*(Q*A+P*B)-A0(I)*YZZ(K)+YP*XZZ(K)
 DGI=G(I+1,J,L)-G(I-1,J,L)
 DGJ=G(I,J+1,L)-GK1(I,J-1)
 DGK=G(I,J,L+1)-GK1(I,J)
 DGII=G(I+1,J,L)-G(I,J,L)-G(I,J,L)+G(I-1,J,L)+A3(I)*DGI
 DGJJ=G(I,J+1,L)-G(I,J,L)-G(I,J,L)+G(I,J-1,L)-B3(J)*DGJ
 DGKK=G(I,J,L+1)-G(I,J,L)-G(I,J,L)+G(I,J,L-1)+C3(K)*DGK
 DGIJ=G(I+1,J+1,L)-G(I-1,J+1,L)-G(I+1,J-1,L)+G(I-1,J-1,L)
```

```
      DGIK=G(I+1,J,L+1)-G(I+1,J,L-1)-G(I-1,J,L+1)+G(I-1,J,L-1)
      DGJK=G(I,J+1,L+1)-G(I,J-1,L+1)-G(I,J+1,L-1)+G(I,J-1,L-1)
      GX=A1(I)*DGI
      GY=-B1(J)*DGJ
      U=GX-SX(I)*GY+CA*AO(I)+SA*YP
      V=GY+SA*AO(I)-CA*YP
      W=RO(I)*(C1(K)*DGK-SZ(I)*GY+SYAW+CA*XZ(K)+SA*YZ(K)+H*(U*AZ+V*BZ))
      AU=U+W*AZ
      AV=V+W*BZ
      QXY=H*(U*U+V*V)
      QQ=QXY+W*W
      AA=DIM(AAO,.2*QQ)
      HZ=AZ*SX(I)-BZ+FH*SZ(I)
      FXX=1.+H*AZ*AZ
      FYY=1.+SX(I)*SX(I)+H*HZ*HZ
      FXY=SX(I)+H*AZ*HZ
      BV=AV-AU*SX(I)-FH*W*SZ(I)
      UU=H*AU*AU
      VV=H*BV*BV
      WW=FH*W*W
      UV=H*AU*BV
      UW=AU*W
      VW=BV*W
      AXX=R1(I)*(FXX*AA-UU)
      AZZ=FH*AA-WW
      AXZ=(RO(I)+RO(I))*(AZ*AA-UW)
      R=-(AXX*SXX(I)+AZZ*SZZ(I)+AXZ*SXZ(I))*GY+T1*(RO(I)*AA*(CZ*GX+(DZ-S
     1X(I)*CZ)*GY)-H*(CA*(AU*AU-AV*AV)+(SA+SA)*AU*AV-QXY*(U*AO(I)+V*YP+(
     2W+W)*(AO(I)*AZ+YP*BZ)))-WW*(CA*XZZ(K)+SA*YZZ(K))-W*W*(U*CZ+V*DZ))
      AXT=ABS(AU*A1(I))
      AYT=ABS(BV*B1(J))
      AZT=ABS(FH*W*C1(K))
      A=RO(I)*BETA*AA/AMAX1(AXT,AYT,AZT,(1.-RO(I)))
      AXT=A*AXT
      AYT=A*AYT
      AZT=A*AZT
      IF (QQ.GE.AA) GO TO 30
      AXX=AXX*A2(I)
      AYY=(FYY*AA-VV)*B2(J)
      AZZ=AZZ*C2(K)
      AXY=-R1(I)*(FXY*AA+UV)*(AB+AB)
      AXZ=AXZ*AC
      AYZ=-RO(I)*(HZ*AA+VW)*(BC+BC)
      BP=AXX
      BM=AXX
      B=-AXX-AXX-Q1*(AYY+AZZ)
      R=AXX*DGII+AYY*DGJJ+AZZ*DGKK+AXY*DGIJ+AYZ*DGJK+AXZ*DGIK+R
      GO TO 40
   30 NS=NS+1
      S=SIGN(1.,U)
      IM=I-IFIX(S)
      IMM=IM-IFIX(S)
```

```
      AXX=UU*A2(I)
      AYY=VV*B2(J)
      AZZ=WW*C2(K)
      AXY=8.*S*UV*AB
      AXZ=8.*S*UW*AC
      AYZ=8.*VW*BC
      BXX=(FXX*QQ-UU)*A2(I)
      BYY=(FYY*QQ-VV)*B2(J)
      BZZ=(FH*QQ-WW)*C2(K)
      BXY=-(FXY*QQ+UV)*(AB+AB)
      BXZ=(AZ*QQ-UW)*(AC+AC)
      BYZ=-(HZ*QQ+VW)*(BC+BC)
      AQ=AA/QQ
      DELTAG=BXX*DGII+BYY*DGJJ+BZZ*DGKK+BXY*DGIJ+BYZ*DGJK+BXZ*DGIK
      DGII=G(I,J,L)-G(IM,J,L)-G(IM,J,L)+G(IMM,J,L)+A3(I)*DGI
      DGJJ=G(I,J,L)-G(I,J-1,L)-G(I,J-1,L)+GK1(I,J-2)-B3(J)*DGJ
      DGKK=G(I,J,L)-G(I,J,L-1)-G(I,J,L-1)+GK2(I,J)+C3(K)*DGK
      DGIJ=G(I,J,L)-G(IM,J,L)-G(I,J-1,L)+G(IM,J-1,L)
      DGIK=G(I,J,L)-G(I,J,L-1)-G(IM,J,L)+G(IM,J,L-1)
      DGJK=G(I,J,L)-G(I,J,L-1)-G(I,J-1,L)+G(I,J-1,L-1)
      GSS=AXX*DGII+AYY*DGJJ+AZZ*DGKK+AXY*DGIJ+AYZ*DGJK+AXZ*DGIK
      B=.5*(AQ-1.)*(AXX+AXX+AXY+AXZ)
      BP=AQ*BXX-(1.-S)*B
      BM=AQ*BXX-(1.+S)*B
      B=-AQ*(BXX+BXX+Q2*(BYY+BZZ))+(AQ-1.)*(2.*(AXX+AYY+AZZ)+AXY+AYZ+AXZ
     1)
      R=(AQ-1.)*GSS+AQ*DELTAG+R
   40 IF (ABS(R).LE.ABS(FR)) GO TO 50
      FR=R
      IR=I
      JR=J
      KR=K
   50 R=R-AYT*(GK1(I,J-1)-G(I,J-1,L))-AZT*(GK1(I,J)-G(I,J,L-1))
      B=B -AXT-AYT-AZT
      BM=BM+AXT
      B=1./(B-BM*C(I-1))
      C(I)=B*BP
   60 D(I)=B*(R-BM*D(I-1))
      CG=0.
      I=I3
      DO 80 M=I1,I3
      CG=D(I)-C(I)*CG
      IF (ABS(CG).LE.ABS(DG)) GO TO 70
      DG=CG
      IG=I
      JG=J
      KG=K
   70 GK2(I,J)=GK1(I,J)
      GK1(I,J)=G(I,J,L)
      G(I,J,L)=G(I,J,L)-CG
   80 I=I-1
      J=J+1
```

```
      IF (J-KY) 20,90,110
   90 IF (I2.GT.ITE2(K)) I3=ITE2(K)
      IF (ITE2(K).EQ.MX) I3=LX
      DO 100 I=I1,I3
      LV=IABS(1-IABS(IV(I,K)))
      RO(I)=AMINO(LV,IABS(IV(I,K)))
  100 R1(I)=LV
      GO TO 20
  110 N=NO
      I=LX+1
      IF (K.LT.KTE1.OR.K.GT.KTE2) GO TO 130
      IO=NX+2-I3
      DO 120 I=IO,I3
      A=1.-RO(I)+AO(I)*AO(I)+SO(I,K)*SO(I,K)
      H=RO(I)/A
      FH=RO(I)*A
      AZ=-AO(I)*XZ(K)-SO(I,K)*YZ(K)
      BZ=-AO(I)*YZ(K)+SO(I,K)*XZ(K)
      HZ=AZ*SX(I)-BZ+FH*SZ(I)
      FYY=1.+SX(I)*SX(I)+H*HZ*HZ
      FXY=SX(I)+H*AZ*HZ
      DGI=G(I+1,KY,L)-G(I-1,KY,L)
      DGK=G(I,KY,L+1)-GK2(I,KY)
      V=SA*AO(I)-CA*SO(I,K)
      U=A1(I)*DGI+CA*AO(I)+SA*SO(I,K)
      W=C1(K)*DGK+SYAW+CA*XZ(K)+SA*YZ(K)
  120 G(I,KY+1,L)=G(I,KY-1,L)+(V*(1.-H*BZ*HZ)-U*FXY-W*HZ)/(FYY*B1(KY))
      I=IO
      IF (IO.NE.ITE1(K)) GO TO 130
      E=G(I3,KY,L)-G(IO,KY,L)
      NO=NO+1
      EO(NO)=EO(NO)+P3*(E-EO(NO))
      N=NO
  130 IF (I.LE.I1) GO TO 170
      I=I-1
      E=0.
      IF (IV(I,K).NE.1) GO TO 160
      ZZ=Z(K)-TYAW*(XC(K)+S1*AO(I)*AO(I))
  140 IF (ZZ.GE.ZO(N-1)) GO TO 150
      N=N-1
      GO TO 140
  150 R=(ZZ-ZO(N-1))/(ZO(N)-ZO(N-1))
      E=R*EO(N)+(1.-R)*EO(N-1)
  160 M=NX+2-I
      G(I,KY+1,L)=G(M,KY-1,L)-E
      G(M,KY+1,L)=G(I,KY-1,L)+E
      GK2(M,KY)=GK1(M,KY)
      GK1(M,KY)=G(M,KY,L)
      G(M,KY,L)=G(I,KY,L)+E
      GO TO 130
C     ACCURATE TRUNCATED BOUNDARY CONDITIONS
  170 CONTINUE
```

```
      DO 180 I=2,NX
  180 G(I,J1-1,L)=(1.-BETY/BSCAL1)*G(I,J1,L)
      DO 190 J=1,MY
      G(I1-1,J,L)=(1.-BETX/BSCAL)*G(I1,J,L)
  190 G(I2+1,J,L)=(1.-BETX/BSCAL)*G(I2,J,L)
      RETURN
      END



      SUBROUTINE VELO (K,L,SV,SM,CP,X,Y,UC,VC,WC)
C     CALCULATES SURFACE VELOCITY
      COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
     1,ITE2(15),AO(129),A1(129),A2(129),A3(129),BO(12),B1(12),B2(12),B3(
     212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
     3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
     4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
     515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
     6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,F00,F01,F10,F11,
     7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
      DIMENSION SV(1), SM(1), CP(1), X(1), Y(1), UC(1), VC(1), WC(1)
      DIMENSION Q2(129), Q2S(129), Q2SX(129), Q2SZ(129), Q2M(129), Q2SM(
     1129), Q2SXM(129), Q2SZM(129), Q2P(129)
      DIMENSION DY(129), DYK(129)
      COMMON /DIM/ NX1,NY1,NZ1,FDIM
      I10=ITE1(KTE1)
      I20=ITE2(KTE1)
      DO 10 KDUM=KTE1,KTE2
      I10=MINO(I10,ITE1(KDUM))
   10 I20=MAXO(I20,ITE2(KDUM))
      J=NY+1
      Q1=.2*FMACH**2
      T1=1./(.7*FMACH**2)
      DO 20 I=I10,I20
      FH=AO(I)*AO(I)+SO(I,K)*SO(I,K)
      H=0.
      IF (IV(I,K).NE.0) H=1./FH
      AZ=-AO(I)*XZ(K)-SO(I,K)*YZ(K)
      BZ=-AO(I)*YZ(K)+SO(I,K)*XZ(K)
      DSI=SO(I+1,K)-SO(I-1,K)
      DSK=SO(I,K+1)-SO(I,K-1)
      SX=A1(I)*DSI
      SZ=C1(K)*DSK
      DGI=G(I+1,J,L)-G(I-1,J,L)
      DGJ=G(I,J+1,L)-G(I,J-1,L)
      DGK=G(I,J,L+1)-G(I,J,L-1)
      U=A1(I)*DGI+SX*B1(J)*DGJ+CA*AO(I)+SA*SC(I,K)
```

```
      V=-B1(J)*DGJ+SA*AO(I)-CA*SO(I,K)
      W=C1(K)*DGK+SZ*B1(J)*DGJ+SYAW+CA*XZ(K)+SA*YZ(K)+H*(U*AZ+V*BZ)
      QQ=H*(U*U+V*V)+W*W
      SV(I)=SIGN(SQRT(QQ),U)
      IF (IV(I,K).EQ.0) SV(I)=SV(I-1)+SV(I-1)-SV(I-2)
      UC(I)=0.
      VC(I)=0.
      WC(I)=0.
      QQ=1.+Q1*(1.-QQ)
      SM(I)=FMACH*SV(I)/SQRT(QQ)
      CP(I)=T1*(QQ**3.5-1.)
      X(I)=XC(K)+.5*SCAL*(AO(I)*AO(I)-SO(I,K)*SO(I,K))
      Y(I)=YC(K)+SCAL*AO(I)*SO(I,K)
      IF (NDES.LE.0) GO TO 20
      W2S=2.*W*H*((-U*YZ(K)+V*XZ(K)+SA*AZ-CA*BZ)-2.*SO(I,K)*H*(U*AZ+V*BZ
     1))
      Q2S(I)=-2.*H*((SA*U-CA*V)-SO(I,K)*H*(U*U+V*V))-W2S
      Q2SX(I)=-2.*B1(J)*DGJ*H*(U+AZ*W)
      Q2SZ(I)=-2.*B1(J)*DGJ*W
   20 CONTINUE
      IF (NDES.LE.0) RETURN
      CALL SURMOD (K,L,SV,SM,CP,X,Y,UC,VC,WC,Q2S,Q2SX,Q2SZ)
      RETURN
      END


      SUBROUTINE SURMOD (K,L,SV,SM,CP,X,Y,UC,VC,WC,Q2S,Q2SX,Q2SZ)
C     PERFORMS SURFACE MODIFICATION IN DESIGN MODE
      COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
     1,ITE2(15),AO(129),A1(129),A2(129),A3(129),BO(12),B1(12),B2(12),B3(
     212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
     3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
     4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
     515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
     6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,FOO,FO1,F10,F11,
     7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
      DIMENSION SV(1), SM(1), CP(1), X(1), Y(1), UC(1), VC(1), WC(1)
      DIMENSION Q2(129), Q2S(129), Q2SX(129), Q2SZ(129), Q2M(129), Q2SM(
     1129), Q2SXM(129), Q2SZM(129), Q2P(129)
      DIMENSION DY(129), DYK(129)
      COMMON /DIM/ NX1,NY1,NZ1,FDIM
      I1=ITE1(K)
      I2=ITE2(K)
      I10=ITE1(KTE1)
      I20=ITE2(KTE1)
      DO 10 KDUM=KTE1,KTE2
```

```
      I10=MINO(I10,ITE1(KDUM))
   10 I20=MAXO(I20,ITE2(KDUM))
      DX=1./FLOAT(NX1)
      CUTLO=.80
      J=NY+1
      Q1=.2*FMACH**2
      T1=1./(.7*FMACH**2)
      KFLAG=0
      IF (K.GT.KTE1) GO TO 30
      DO 20 I=I10,I20
      DYK(I)=0.
      Q2(I)=QPRE(I,K)*QPRE(I,K)-SV(I)*SV(I)
      UC(I)=QPRE(I,K)
      VC(I)=SO(I,K)-SOPRE(I,K)
      Q2SM(I)=Q2S(I)
      Q2SXM(I)=Q2SX(I)
   20 Q2SZM(I)=Q2SZ(I)
      RETURN
   30 DO 40 I=I10,I20
      Q2P(I)=QPRE(I,K)*QPRE(I,K)-SV(I)*SV(I)
      UC(I)=QPRE(I,K)
   40 VC(I)=SO(I,K)-SOPRE(I,K)
      IF (K.GT.KTE1+1) GO TO 60
      DO 50 I=I10,I20
   50 Q2M(I)=Q2P(I)
   60 CONTINUE
      DT=TSTEP*DX
      KM=K-1
      IF (KFLAG.EQ.1) KM=KTE2
      I=I20
      DY(I)=0.
      DYK(I)=0.
   70 I=I-1
      DY(I)=0.
      IF (I.LE.I10) GO TO 100
      IP=I+1
      IM=I-1
      IF (I.GT.ITE2(KM).OR.I.LT.ITE1(KM)) GO TO 70
      Q2X=2.*A1(I)*(Q2(IP)-Q2(I))
      IF (Q2SXM(I).LE.0.) Q2X=2.*A1(I)*(Q2(I)-Q2(IM))
      Q2Z=2.*C1(KM)*(Q2P(I)-Q2(I))
      IF (Q2SZM(I).LE.0.) Q2Z=2.*C1(KM)*(Q2(I)-Q2M(I))
      DS=Q2(I)+.5*DT*(Q2SM(I)*Q2(I)+Q2SXM(I)*Q2X+Q2SZM(I)*Q2Z)
      FF00=F00
      FF01=F01
      FF10=F10
      IF (ABS(SM(I)).LE.CUTLO) FF10=0.
      FF11=F11
      FAC=1./(FF00+FF01-FF10-FF11)
      DY(I)=(DT*DS-DY(IP)*(FF10+FF11)+DYK(I)*(FF01-FF11)+DYK(IP)*FF11)*F
     1AC
      DUMM=SO(I,KM)+DY(I)
```

```
      IF (DUMM.LT.SOPRE(I,KM)) DUMM=SOPRE(I,KM)
      DY(I)=DUMM-SO(I,KM)
      SO(I,KM)=DUMM
      IF (SO(I,KM).LE.SOPRE(I,KM)) GO TO 90
      IF (ABS(DS).LT.RDSO) GO TO 80
      RDSO=AMAX1(RDSO,ABS(DS))
      IQ=I
      KQ=KM
   80 CONTINUE
      NDQ=NDQ+1
      RDQ=RDQ+ABS(DS)
   90 CONTINUE
      GO TO 70
  100 CONTINUE
      IF (KFLAG.EQ.1) RETURN
      DO 110 I=I10,I20
      DYK(I)=DY(I)
      Q2M(I)=Q2(I)
      Q2(I)=Q2P(I)
      Q2SM(I)=Q2S(I)
      Q2SXM(I)=Q2SX(I)
      Q2SZM(I)=Q2SZ(I)
  110 CONTINUE
      IF (K.LT.KTE2) RETURN
      DO 120 I=I10,I20
  120 Q2P(I)=AMIN1(0.,2.*Q2(I)-Q2M(I))
      KFLAG=1
      GO TO 60
      END


      SUBROUTINE DRAGC (IND,SCALX)
C     COMPUTES THE WAVE DRAG BY VOLUME INTEGRATION OF ENTROPY INEQUALITY
      COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
     1,ITE2(15),AO(129),A1(129),A2(129),A3(129),BO(12),B1(12),B2(12),B3(
     212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
     3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
     4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
     515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
     6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,F00,F01,F10,F11,
     7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
      COMMON /FLO/ STRIP,P1,P2,P3,BETA,FR,IR,JR,KR,DG,IG,JG,KG,NS,FSWEEP
      COMMON /SWP/ DXYZ(129),GK1(129,15),GK2(129,15),SX(129),SZ(129),SXX
     1(129),SXZ(129),SZZ(129),RO(129),R1(129),C(129),D(129),G10(15),G20(
     215),G30(15),G40(15),G1(15),G2(15),I1,I2,K,L,NO,LX,MX,KY,MY,T1,AA0,
     3Q1,Q2,TYAW,S1
```

```
      DIMENSION XXX(34), YYY(34), DRAG(34)
      COMMON /DIM/ NX1,NY1,NZ1,FDIM
      IF (SCALX.EQ.0.0) SCALX=5./(Z(KTE2)-Z(KTE1))
      TTTX=3.
      SSSX=-SCALX*XC(KTE1)
      DX=2./FLOAT(NX1)
      DZO=1./FLOAT(NZ1)
      DVOL=2./FLOAT(NX1*NY1*NZ1)
      DO 10 K=KTE1,KTE2
   10 SECDRG(K)=0.
      FACZ=.5
      LX=NX/2+1
      PI=3.1415927
      RAD=PI/180.
      ANG=-5.*RAD
      NDARK=40
      SCUT=.02
      INDPLT=0
      SZO=200.*(FLOAT(NX)/156.)**2
      DO 130 K=KTE1,KTE2
      INDPLT=INDPLT+1
      DO 20 J=2,KY
      XXX(J)=0.
      YYY(J)=0.
   20 DRAG(J)=0.
      SSSY=5.*(Z(K)-Z(KTE1))/(Z(KTE2)-Z(KTE1))+2.45
      KP=K+1
      KM=K-1
      I1=ITE1(K)
      I2=ITE2(K)
      DO 50 I=I1,I2
      IDRGPLT(I,K)=-1
      IP=I+1
      IM=I-1
      SX(I)=A1(I)*(SO(IP,K)-SO(IM,K))
      SZ(I)=C1(K)*(SO(I,KP)-SO(I,KM))
      FACY=1.
      DO 40 J=2,KY
      IF (J.EQ.KY) FACY=.5
      YP=SO(I,K)+BO(J)
      FH=AO(I)*AO(I)+YP*YP
      H=1./FH
      AZ=-AO(I)*XZ(K)-YP*YZ(K)
      BZ=-AO(I)*YZ(K)+YP*XZ(K)
      DGI=G(IP,J,K)-G(IM,J,K)
      DGJ=G(I,J+1,K)-G(I,J-1,K)
      DGK=G(I,J,KP)-G(I,J,KM)
      GX=A1(I)*DGI
      GY=-B1(J)*DGJ
      U=GX-SX(I)*GY+CA*AO(I)+SA*YP
      V=GY+SA*AO(I)-CA*YP
      W=(C1(K)*DGK-SZ(I)*GY+SYAW+CA*XZ(K)+SA*YZ(K)+H*(U*AZ+V*BZ))
```

```
      AU=U+W*AZ
      QXY=H*(U*U+V*V)
      QQ=QXY+W*W
      AA=DIM(AAO,.2*QQ)
      DUMM=0.
      IF (QQ.LT.AA) GO TO 30
      UU=H*AU*AU
      AXX=UU*A2(I)
      AQ=AA/QQ
      XJACO=1.*SCALZ*((1.+4.*BO(J)*BO(J))**1.5)*DVOL/SCAL
      DRGSS=(G(I+1,J,K)-2.*G(I,J,K)+G(I-1,J,K))/(DX*DX)
      DRGSS=.5*(DRGSS-ABS(DRGSS))
      ROCS=FMACH*FMACH*(FMACH*FMACH*AA)**1.5
      DUMM=2.*DX*SCAL*SCAL*(1.-AQ)*ROCS*AXX*DRGSS*DRGSS*SQRT(H)*XJACO/AW
     1ING
C     SECOND ORDER ACCURATE VOLUME INTEGRAL
      VOLDRG=VOLDRG+FACY*FACZ*DUMM
      SECDRG(K)=SECDRG(K)+AWING*DUMM/(SCALZ*DZO)
      IF (J.EQ.KY) IDRGPLT(I,K)=1000000.*DUMM
   30 IF (IND.NE.1) GO TO 40
      IF (I.LT.LX) GO TO 40
      XX=XC(K)+.5*SCAL*(AO(I)*AO(I)-(SO(I,K)+BO(J))**2)
      YY=YC(K)+SCAL*AO(I)*(SO(I,K)+BO(J))
      DRAG(J)=DRAG(J)+DUMM
      XXX(J)=XXX(J)+DUMM*XX
      YYY(J)=YYY(J)+DUMM*YY
   40 CONTINUE
   50 CONTINUE
      IF (IND.NE.1) GO TO 130
      DO 60 J=2,KY
      IF (DRAG(J).LT.1.E-50) GO TO 60
      XXX(J)=XXX(J)/DRAG(J)
      YYY(J)=YYY(J)/DRAG(J)
   60 CONTINUE
      IPREV=0
      DO 120 JJ=2,KY
      J=KY+2-JJ
      SIZE=SZO*DRAG(J)
      IF (SIZE.LT.SCUT) SIZE=0.
      XCD=SCALX*XXX(J)+SSSX+TTTX
      YCD=SCALX*YYY(J)+SSSY
      XL=XCD-SIZE*COS(ANG)
      YL=YCD-SIZE*SIN(ANG)
      IF (SIZE.LT.SCUT.AND.IPREV.EQ.0) GO TO 110
      IF (J.EQ.KY) GO TO 110
      IF (IPREV.EQ.1.AND.SIZE.GE.SCUT) GO TO 80
      IF (SIZE.LT.SCUT) GO TO 70
      XX=2.*(XXX(J)-XC(K))/SCAL
      YY=2.*(YYY(J)-YC(K))/SCAL
      RR=SQRT(SQRT(XX*XX+YY*YY))
      THET=.5*ATAN(YY/XX)
      XX=RR*COS(THET)
```

```
      YY=RR*SIN(THET)
      YY=YY-BO(J)+BO(J+1)
      XCDO=XC(K)+.5*SCAL*(XX*XX-YY*YY)
      YCDO=YC(K)+SCAL*XX*YY
      XCDO=SCALX*XCDO+SSSX+TTTX
      YCDO=SCALX*YCDO+SSSY
      XLO=XCDO
      YLO=YCDO
      GO TO 80
   70 XX=2.*(XXX(J+1)-XC(K))/SCAL
      YY=2.*(YYY(J+1)-YC(K))/SCAL
      RR=SQRT(SQRT(XX*XX+YY*YY))
      THET=.5*ATAN(YY/XX)
      XX=RR*COS(THET)
      YY=RR*SIN(THET)
      YY=YY-BO(J+1)+BO(J)
      XCD=XC(K)+.5*SCAL*(XX*XX-YY*YY)
      YCD=YC(K)+SCAL*XX*YY
      XCD=SCALX*XCD+SSSX+TTTX
      YCD=SCALX*YCD+SSSY
      XL=XCD
      YL=YCD
   80 CONTINUE
      IF (KTE2.LT.10) GO TO 90
      IF (MOD(INDPLT,2).EQ.0) GO TO 110
   90 CONTINUE
      DO 100 L=1,NDARK
      FAC=FLOAT(L)/FLOAT(NDARK)
      XX=FAC*XCD+(1.-FAC)*XCDO
      YY=FAC*YCD+(1.-FAC)*YCDO
      CALL PLOT (XX,YY,3)
      XX=FAC*XL+(1.-FAC)*XLO
      YY=FAC*YL+(1.-FAC)*YLO
  100 CALL PLOT (XX,YY,2)
  110 IPREV=0
      IF (SIZE.GE.SCUT) IPREV=1
      XCDO=XCD
      YCDO=YCD
      XLO=XL
      YLO=YL
  120 CONTINUE
  130 CONTINUE
      FACZ=1.
      RETURN
      END
```

```
      SUBROUTINE CPLOT (I1,I2,X,Y,Z,A,B,C,D,FMACH)
C     PLOTS CP AT EQUAL INTERVALS IN THE MAPPED PLANE
      DIMENSION KODE(3), LINE(100), X(1), Y(1), A(1), B(1), C(1), D(1),
     1Z(1)
      DATA KODE/1H ,1H+,1HO/
      IWRIT=6
      AAO=.2+1./FMACH**2
      WRITE (IWRIT,80)
      DO 10 I=1,100
   10 LINE(I)=KODE(1)
      I10=I1
      I20=I2
      ISPA=1
      LX=(I1+I2)/2
      FDEN=1./B(I20)
      LXO=.85*FLOAT(LX)
      IF (LXO.LT.I2-49) ISPA=2
      AMAX=0.
      CMAX=0.
      DO 20 I=I10,I20
      AMAX=AMAX1(AMAX,ABS(X(I)))
      AMAX=AMAX1(AMAX,ABS(Y(I)))
      CMAX=AMAX1(CMAX,ABS(C(I)))
   20 CMAX=AMAX1(CMAX,ABS(D(I)))
      DO 70 I=LXO,I2,ISPA
      XFRAC=FDEN*B(I)
      Y(I)=SQRT(Y(I)**2/(AAO-.2*Y(I)**2))
      K1=(59./AMAX)*ABS(X(I))+41.
      K1=MINO(K1,100)
      LINE(K1)=KODE(2)
      K2=(59./AMAX)*ABS(Y(I))+41.
      K2=MINO(K2,100)
      LINE(K2)=KODE(3)
      K3=(36./CMAX)*D(I)+1.
      K3=MINO(K3,40)
      IF (K3.GE.1) GO TO 30
      K3=1.
      GO TO 40
   30 LINE(K3)=KODE(2)
   40 K4=(36./CMAX)*C(I)+1.
      IF (K4.GE.1) GO TO 50
      K4=1
      GO TO 60
   50 LINE(K4)=KODE(3)
   60 CONTINUE
      JJ=0
      IF (Z(I).GT.0) JJ=1
      WRITE (IWRIT,90) XFRAC,X(I),Y(I),JJ,LINE
      LINE(K1)=KODE(1)
      LINE(K2)=KODE(1)
      LINE(K3)=KODE(1)
      LINE(K4)=KODE(1)
```

```
 70 CONTINUE
    I1=I10
    I2=I20
    RETURN
C
 80 FORMAT (1X,/,3X,3HX/C,4X,5HMCOMP,4X,5HMDSGN,3H ON)
 90 FORMAT (1X,F5.2,2F9.3,I3,2X,100A1)
    END
```

```
    SUBROUTINE FORCF (I1,I2,X,Y,CP,AL,CHORD,XM,CL,CD,CM)
C   CALCULATES SECTION FORCE COEFFICIENTS
    DIMENSION X(1), Y(1), CP(1)
    RAD=57.2957795130823
    ALPHA=AL/RAD
    CL=0.
    CD=0.
    CM=0.
    N=I2-1
    DO 10 I=I1,N
    DX=(X(I+1)-X(I))/CHORD
    DY=(Y(I+1)-Y(I))/CHORD
    XA=(.5*(X(I+1)+X(I))-XM)/CHORD
    YA=.5*(Y(I+1)+Y(I))/CHORD
    CPA=.5*(CP(I+1)+CP(I))
    DCL=-CPA*DX
    DCD=CPA*DY
    CL=CL+DCL
    CD=CD+DCD
 10 CM=CM+DCD*YA-DCL*XA
    DCL=CL*COS(ALPHA)-CD*SIN(ALPHA)
    CD=CL*SIN(ALPHA)+CD*COS(ALPHA)
    CL=DCL
    RETURN
    END
```

```
    SUBROUTINE TOTFOR (KTE1,KTE2,CHORD,SCL,SCD,SCM,Z,XC,CL,CD,CMP,CMR,
   1CMY,AWING)
C   CALCULATES TOTAL FORCE COEFFICIENTS
```

```
      DIMENSION CHORD(1), SCL(1), SCD(1), SCM(1), Z(1), XC(1)
      SPAN=Z(KTE2)-Z(KTE1)
      CL=0.
      CD=0.
      CMP=0.
      CMR=0.
      CMY=0.
      S=0.
      N=KTE2-1
      DO 10 K=KTE1,N
      DZ=.5*(Z(K+1)-Z(K))
      AZ=.5*(Z(K+1)+Z(K))
      CL=CL+DZ*(SCL(K+1)*CHORD(K+1)+SCL(K)*CHORD(K))
      CD=CD+DZ*(SCD(K+1)*CHORD(K+1)+SCD(K)*CHORD(K))
      CMP=CMP+DZ*(CHORD(K+1)*(SCM(K+1)*CHORD(K+1)-SCL(K+1)*XC(K+1))+CHOR
     1D(K)*(SCM(K)*CHORD(K)-SCL(K)*XC(K)))
      CMR=CMR+AZ*DZ*(SCL(K+1)*CHORD(K+1)+SCL(K)*CHORD(K))
      CMY=CMY+AZ*DZ*(SCD(K+1)*CHORD(K+1)+SCD(K)*CHORD(K))
   10 S=S+DZ*(CHORD(K+1)+CHORD(K))
      AWING=S
      CL=CL/S
      CD=CD/S
      CMP=CMP*SPAN/S**2
      CMR=(CMR+CMR)/(S*SPAN)
      CMY=(CMY+CMY)/(S*SPAN)
      RETURN
      END



      SUBROUTINE OPT (QQ1,QQ2,QQ3,QQ4)
C     INITIALIZES PARAMETERS FOR THE OPTIMIZATION ROUTINE
C     AND CALLS OPTIMIZER
      DIMENSION QQ1(1), QQ2(1), QQ3(1), QQ4(1)
      DIMENSION X(20), G(20), H(20,20), W(60), XM(20)
      EXTERNAL DRFCT
      IWRIT=6
      N=4
      DFN=.0003
      HH=1.
      MODE=1
      DO 10 I=1,N
   10 XM(I)=.05
      X(1)=QQ2(2)
      X(2)=QQ3(2)
      X(3)=QQ2(3)
      X(4)=QQ3(3)
```

```
      MAXFN=30
      IPRINT=1
      EPS=.1
      CALL VA10A (DRFCT,N,X,F,G,H,W,DFN,XM,HH,EPS,MODE,MAXFN,IPRINT,IEXI
     1T)
      RETURN
      END




      SUBROUTINE DRFCT (NDUM,XDUM,F)
C     EVALUATES THE DRAG AS A FUNCTION OF THE SPEED DISTRIBUTION
C     DETERMINED BY THE OPTIMIZER
      COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
     1,ITE2(15),AO(129),A1(129),A2(129),A3(129),BO(12),B1(12),B2(12),B3(
     212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
     3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
     4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
     515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
     6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,F00,F01,F10,F11,
     7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
      COMMON /FLO/ STRIP,P1,P2,P3,BETA,FR,IR,JR,KR,DG,IG,JG,KG,NS,FSWEEP
      DIMENSION XDUM(1)
      DIMENSION SV(129), SM(129), CP(129), X(129), Y(129), UC(129), VC(1
     129), WC(129), CHORD(15), SCL(15), SCD(15), SCM(15)
      DATA ISTART/1/,NITOT/0/
      DATA VAR/0.0/
      IWRIT=6
      AAO=1./FMACH**2+.2
      NE=129
      LX=NX/2+1
      QCUT=.015
      QMCUT=.09
      IMAX=150
      QQ2(2)=XDUM(1)
      QQ3(2)=XDUM(2)
      QQ2(3)=XDUM(3)
      QQ3(3)=XDUM(4)
      IF (ISTART.NE.1) CALL TREAD
      ISTART=ISTART+1
      IF (ISTART.GE.10) IMAX=200
      DO 10 KL=1,5
      PRINT 70, ZQSTA(KL),QQ1(KL),QQ2(KL),QQ3(KL),QQ4(KL)
   10 CONTINUE
      CALL SETQS (NE,NX,QPRE,SO,SOPRE,ITE1,ITE2,KTE1,KTE2,Z,ZQSTA,AO,PCQ
     11,PCQ2,PCQ3,UC,VC,QQ1,QQ2,QQ3,QQ4,PCS1,PCS2,DSURF,NQSTA)
      WRITE (IWRIT,120)
```

```
      WRITE (IWRIT,80)
      ITER=0
   20 ITER=ITER+1
      CALL MIXFLO
      NITOT=NITOT+1
      VOLDRG=0.
      CALL DRAGC (0)
      RDSO=0.
      NDQ=0
      RDQ=0.
      DO 30 K=KTE1,KTE2
      CALL VELO (K,K,SV,SM,CP,X,Y,UC,VC,WC)
      I1=ITE1(K)
      I2=ITE2(K)
      CHORD(K)=X(I1)-X(LX)
      CALL FORCF (I1,I2,X,Y,CP,AL,CHORD(K),XC(K),SCL(K),SCD(K),SCM(K))
   30 CONTINUE
      CALL TOTFOR (KTE1,KTE2,CHORD,SCL,SCD,SCM,Z,XC,CL,CD1,CMP,CMR,CMY,A
     1WING)
      IF (NDQ.GT.0) RDQ=RDQ/FLOAT(NDQ)
      WRITE (IWRIT,90) ITER,FR,DG,NS,RDQ,RDSO,VOLDRG,CL,NITOT
      IF (ITER.GE.IMAX) GO TO 40
      IF (AMAX1(RDQ,RDSO).GE.2.) GO TO 40
      IF (RDQ.GT.QCUT.OR.RDSO.GT.QMCUT) GO TO 20
   40 VOLDRG=0.
      CALL DRAGC (0)
      WRITE (IWRIT,100) VOLDRG
      NDUMO=NDUM/2
      DO 50 I=2,3
      XMA1=SQRT(QQ2(I)**2/(AA0-.2*QQ2(I)**2))
      XMA2=SQRT(QQ3(I)**2/(AA0-.2*QQ3(I)**2))
   50 WRITE (IWRIT,110) I,QQ2(I),QQ3(I),XMA1,XMA2
      DO 60 K=KTE1,KTE2
      CALL VELO (K,K,SV,SM,CP,X,Y,UC,VC,WC)
      I1=ITE1(K)
      I2=ITE2(K)
      CHORD(K)=X(I1)-X(LX)
      CALL FORCF (I1,I2,X,Y,CP,AL,CHORD(K),XC(K),SCL(K),SCD(K),SCM(K))
      WRITE (IWRIT,120) Z(K),SCL(K)
      CALL CPLOT (I1,I2,SM,UC,VC,QPRE(1,K),AO,SOPRE(1,K),SO(1,K),FMACH)
   60 CONTINUE
      F=VOLDRG
      CALL CHEKPTX (VAR)
      RETURN
C
   70 FORMAT (1X,5F10.5)
   80 FORMAT (1X,6X,4HITER,5X,5HRESID,6X,4HDPHI,8X,2HNS,5X,5HDQAVE,5X,5H
     1DQMAX,6X,4HDRAG,8X,2HCL,5X,5HNITOT,///)
   90 FORMAT (1X,I10,2E10.3,I10,2E10.3,F10.5,F6.2,I6)
  100 FORMAT (1X,//,1X5HDRAG=,F10.5,20H     SPEED1     SPEED2,20H     MACH1
     1     MACH2   )
  110 FORMAT (1X,/,6X,I10,4F10.5)
```

```
120 FORMAT (1H1,1X,7HZ(K) = ,F10.5,2X,6H CL = ,F10.2//)
    END



    SUBROUTINE TREAD
C   READS THE POTENTIAL STORED AT THE END OF THE LAST LINE SEARCH
    COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
   1,ITE2(15),A0(129),A1(129),A2(129),A3(129),B0(12),B1(12),B2(12),B3(
   212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
   3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
   4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
   515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
   6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,F00,F01,F10,F11,
   7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
    NT=8
    REWIND NT
    READ (NT) NX,NY,NZ,NM,K1,K2,NIT
    MX=NX+1
    MY=NY+2
    MZ=NZ+3
    DO 10 K=1,MZ
    READ (NT) ((G(I,J,K),I=1,MX),J=1,MY)
 10 CONTINUE
    READ (NT) (EO(K),K=K1,K2)
    READ (NT) ((SO(I,K),I=1,MX),K=1,MZ)
    REWIND NT
    RETURN
    END



    SUBROUTINE TWRIT
C   STORES THE POTENTIAL AT COMPLETION OF LINE SEARCH
    COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
   1,ITE2(15),A0(129),A1(129),A2(129),A3(129),B0(12),B1(12),B2(12),B3(
   212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
   3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
   4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
   515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
   6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,F00,F01,F10,F11,
   7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
```

```
      NT=8
      REWIND NT
      NM=0
      NIT=0
      K1=KTE1-1
      K2=KTE2+ITE2(KTE2)-NX/2
      WRITE (NT) NX,NY,NZ,NM,K1,K2,NIT
      MX=NX+1
      MY=NY+2
      MZ=NZ+3
      DO 10 K=1,MZ
      WRITE (NT) ((G(I,J,K),I=1,MX),J=1,MY)
   10 CONTINUE
      WRITE (NT) (EO(K),K=K1,K2)
      WRITE (NT) ((SO(I,K),I=1,MX),K=1,MZ)
      REWIND NT
      PRINT 20
      RETURN
C
   20 FORMAT (1X,15H WRITE ON TAPE8)
      END
```

```
      SUBROUTINE VA10A (FUNCT,N,X,F,G,H,W,DFN,XM,HH,EPS,MODE,MAXFN,IPRIN
     1T,IEXIT)
C     OPTIMIZATION SUBROUTINE
C     PERFORMS LINE SEARCH FOR DRAG MINIMUM
      REAL X(1),G(1),H(1),W(1),XM(1)
      IF (IPRINT.NE.0) PRINT 190
      IF (IPRINT.NE.0) PRINT 200, DFN,HH,(XM(I),I=1,N)
      NN=N*(N+1)/2
      IG=N
      IGG=N+N
      IS=IGG
      IDIFF=1
      IEXIT=0
      IR=N
      IF (MODE.EQ.3) GO TO 40
      IF (MODE.EQ.2) GO TO 30
      IJ=NN+1
      DO 20 I=1,N
      DO 10 J=1,I
      IJ=IJ-1
   10 H(IJ)=0.
   20 H(IJ)=1.
      GO TO 40
```

```
   30 CONTINUE
      CALL MC11B (H,N,IR)
      IF (IR.LT.N) RETURN
   40 CONTINUE
      Z=F
      ITN=0
      CALL FUNCT (N,X,F)
      CALL TWRIT
      IFN=1
      DF=DFN
      IF (DFN.EQ.0.) DF=F-Z
      IF (DFN.LT.0.) DF=ABS(DF*F)
      IF (DF.LE.0.) DF=1.
   50 CONTINUE
      GO TO 170
   60 CONTINUE
      IF (IFN.GE.MAXFN) GO TO 130
      IF (IPRINT.EQ.0) GO TO 70
      IF (MOD(ITN,IPRINT).NE.0) GO TO 70
      PRINT 210, ITN,IFN
      PRINT 220, F
      IF (IPRINT.LT.0) GO TO 70
      PRINT 220, (X(I),I=1,N)
      PRINT 220, (W(IG+I),I=1,N)
   70 CONTINUE
      ITN=ITN+1
      DO 80 I=1,N
   80 W(I)=-W(IG+I)
      CALL MC11E (H,N,W,G,IR)
      Z=0.
      GSO=0.
      DO 90 I=1,N
      W(IS+I)=W(I)
      IF (Z*XM(I).GE.ABS(W(I))) GO TO 90
      Z=ABS(W(I))/XM(I)
   90 GSO=GSO+W(IG+I)*W(I)
      IEXIT=2
      IF (GSO.GE.0.) GO TO 140
      ALPHA=-2.*DF/GSO
      PRINT 230, ALPHA
      IF (ALPHA.GT.1.) ALPHA=1.
      DSTEPO=.03/SQRT(ABS(GSO))
      JMIN=1
      FDMIN=F
      DO 110 JD=2,5
      DSTEP=FLOAT(JD-1)*DSTEPO
      DO 100 I=1,N
  100 W(I)=X(I)+DSTEP*W(IS+I)
      CALL FUNCT (N,W,F1)
      IF (F1.GE.FDMIN) GO TO 110
      FDMIN=F1
      JMIN=JD
```

```
110 CONTINUE
    DSTEP=FLOAT(JMIN-1)*DSTEPO
    DO 120 I=1,N
120 W(I)=X(I)+DSTEP*W(IS+I)
    CALL FUNCT (N,W,F1)
    GO TO 170
130 CONTINUE
    IEXIT=3
    GO TO 150
140 CONTINUE
    IF (IDIFF.EQ.2) GO TO 150
    IDIFF=2
    GO TO 50
150 CONTINUE
    DO 160 I=1,N
160 G(I)=W(IG+I)
    IF (IPRINT.EQ.0) RETURN
    PRINT 210, ITN,IFN,IEXIT
    PRINT 220, F
    PRINT 220, (X(I),I=1,N)
    PRINT 220, (G(I),I=1,N)
    RETURN
170 CONTINUE
    IF (ITN.GE.1) RETURN
    CALL FUNCT (N,X,F)
    IFN=IFN+1
    CALL TWRIT
    DO 180 I=1,N
    Z=HH*XM(I)
    ZZ=X(I)
    X(I)=ZZ+Z
    CALL FUNCT (N,X,F1)
    W(IG+I)=(F1-F)/Z
180 X(I)=ZZ
    IFN=IFN+N
    GO TO 60
C
190 FORMAT (15H1ENTRY TO VA10A,/)
200 FORMAT (6H DFN =,F10.5,5H HH =,F10.5,/,8H XM(I) =,(F10.5))
210 FORMAT (24I5)
220 FORMAT ((8E15.7))
230 FORMAT (1X,7HALPHA =,F10.5)
    END
```

```
      SUBROUTINE MC11B (A,N,IR)
C     OPTIMIZATION SUBROUTINE
C     FACTORIZE A MATRIX GIVEN IN A
      DIMENSION A(1)
      IR=N
      IF (N.GT.1) GO TO 10
      IF (A(1).GT.0.) RETURN
      A(1)=0.
      IR=0
      RETURN
   10 CONTINUE
      NP=N+1
      II=1
      DO 50 I=2,N
      AA=A(II)
      NI=II+NP-I
      IF (AA.GT.0.) GO TO 20
      A(II)=0.
      IR=IR-1
      II=NI+1
      GO TO 50
   20 CONTINUE
      IP=II+1
      II=NI+1
      JK=II
      DO 40 IJ=IP,NI
      V=A(IJ)/AA
      DO 30 IK=IJ,NI
      A(JK)=A(JK)-A(IK)*V
   30 JK=JK+1
   40 A(IJ)=V
   50 CONTINUE
      IF (A(II).GT.0.) RETURN
      A(II)=0.
      IR=IR-1
      RETURN
      END
```

```
      SUBROUTINE MC11E (A,N,Z,W,IR)
C     OPTIMIZATION SUBROUTINE
C     MULTIPLY A VECTOR Z BY THE INVERSE OF THE FACTORS GIVEN IN A
      DIMENSION A(1), Z(1), W(1)
      IF (IR.LT.N) RETURN
      W(1)=Z(1)
      IF (N.GT.1) GO TO 10
```

```
      Z(1)=Z(1)/A(1)
      RETURN
   10 CONTINUE
      DO 30 I=2,N
      IJ=I
      I1=I-1
      V=Z(I)
      DO 20 J=1,I1
      V=V-A(IJ)*Z(J)
   20 IJ=IJ+N-J
      W(I)=V
   30 Z(I)=V
      Z(N)=Z(N)/A(IJ)
      NP=N+1
      DO 50 NIP=2,N
      I=NP-NIP
      II=IJ-NIP
      V=Z(I)/A(II)
      IP=I+1
      IJ=II
      DO 40 J=IP,N
      II=II+1
   40 V=V-A(II)*Z(J)
   50 Z(I)=V
      RETURN
      END



      SUBROUTINE REFIN (ALFO)
C     HALVES MESH SIZE
      COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
     1,ITE2(15),A0(129),A1(129),A2(129),A3(129),B0(12),B1(12),B2(12),B3(
     212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
     3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
     4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
     515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
     6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,FOO,FO1,F10,F11,
     7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
      DIMENSION ALFO(1)
      MX=NX+1
      KY=NY+1
      MY=NY+2
      MZ=NZ+3
      MXO=NX/2+1
      MZO=NZ/2+2
      K=1
```

```
      KK=K
      IF (KSYM.EQ.0) GO TO 10
      MZO=NZ/2+3
      K=2
      KK=K
   10 DO 70 K=KK,MZO
      J=NY/2+1
      JJ=KY
   20 I=MXO
      II=MX
   30 G(II,JJ,K)=G(I,J,K)
      I=I-1
      II=II-2
      IF (I.GT.0) GO TO 30
      J=J-1
      JJ=JJ-2
      IF (J.GT.0) GO TO 20
      DO 40 J=1,KY,2
      DO 40 I=2,NX,2
   40 G(I,J,K)=.5*(G(I+1,J,K)+G(I-1,J,K))
      DO 60 I=1,MX
      DO 50 J=2,NY,2
   50 G(I,J,K)=.5*(G(I,J+1,K)+G(I,J-1,K))
   60 G(I,MY,K)=0.
   70 CONTINUE
      IF (KSYM.NE.0) GO TO 80
      MZM=MZO
      MZST=NZ+1
      GO TO 90
   80 MZM=MZO
      MZST=MZ
   90 CONTINUE
      DO 100 J=1,MY
      DO 100 I=1,MX
  100 G(I,J,MZST)=G(I,J,MZM)
      IF (MZST.EQ.1) GO TO 120
      MZST=MZST-1
      DO 110 J=1,MY
      DO 110 I=1,MX
  110 G(I,J,MZST)=0.5*(G(I,J,MZM)+G(I,J,MZM-1))
      MZM=MZM-1
      MZST=MZST-1
      GO TO 90
  120 CONTINUE
      TYAW=SYAW/CYAW
      S1=.5*SCAL
      NO=KTE1-1
      EO(NO)=0.
      K=2
      KK=K
      IF (KSYM.NE.0) KK=K+1
      DO 200 K=KK,MZ
```

```
      N=NO
      I=MXO+1
      IF (K.LT.KTE1.OR.K.GT.KTE2) GO TO 150
      I1=ITE1(K)
      I2=ITE2(K)
      DO 130 I=I1,I2
      DSI=SO(I+1,K)-SO(I-1,K)
      DSK=SO(I,K+1)-SO(I,K-1)
      SX=A1(I)*DSI
      SZ=C1(K)*DSK
      DGI=G(I+1,KY,K)-G(I-1,KY,K)
      DGK=G(I,KY,K+1)-G(I,KY,K-1)
      R=AMINO(1,IV(I,K))
      A=1.0-R+AO(I)*AO(I)+SO(I,K)*SO(I,K)
      H=R/A
      FH=R*A
      AZ=-AO(I)*XZ(K)-SO(I,K)*YZ(K)
      BZ=-AO(I)*YZ(K)+SO(I,K)*XZ(K)
      HZ=AZ*SX-BZ+FH*SZ
      FYY=1.0+SX*SX+H*HZ*HZ
      FXY=SX+H*AZ*HZ
      U=A1(I)*DGI+CA*AO(I)+SA*SO(I,K)
      W=C1(K)*DGK+SYAW+CA*XZ(K)+SA*YZ(K)
      V=SA*AO(I)-CA*SO(I,K)
  130 G(I,KY+1,K)=G(I,KY-1,K)+(V*(1.0-H*BZ*HZ)-U*FXY-W*HZ)/(FYY*B1(KY))
      NO=NO+1
      EO(NO)=G(I2,KY,K)-G(I1,KY,K)
      N=NO
      I=I1
      IF (K.NE.KTE2.OR.YAW.LE.O.) GO TO 150
  140 I=I+1
      M=NX+2-I
      NO=NO+1
      EO(NO)=G(M,KY,K)-G(I,KY,K)
      IF (I.LT.MXO) GO TO 140
      I=I1
  150 I=I-1
      E=0.
      IF (IV(I,K).NE.1) GO TO 180
      ZZ=Z(K)-TYAW*(XC(K)+S1*AO(I)*AO(I))
  160 IF (ZZ.GE.ZO(N-1)) GO TO 170
      N=N-1
      GO TO 160
  170 R=(ZZ-ZO(N-1))/(ZO(N)-ZO(N-1))
      E=R*EO(N)+(1.-R)*EO(N-1)
  180 M=NX+2-I
      G(I,KY+1,K)=G(M,KY-1,K)-E
      G(M,KY+1,K)=G(I,KY-1,K)+E
      IF (IV(I,K).NE.-1) GO TO 190
      G(I,KY,K)=.5*G(I,KY,K-1)+.25*(G(I,KY,K+1)+G(M,KY,K+1))
      IF (IV(I,K+1).LT.1) G(I,KY,K)=.5*G(I,KY,K+1)+.25*(G(I,KY,K-1)+G(M,
     1KY,K-1))
```

```
      G(M,KY,K)=G(I,KY,K)
      G(I,KY-1,K)=.5*(G(I,KY,K)+G(I,KY-2,K))
      G(M,KY-1,K)=.5*(G(M,KY,K)+G(M,KY-2,K))
  190 IF (I.GT.2) GO TO 150
  200 CONTINUE
      EO(NO+1)=0.
      K2=KTE1+(KTE2-KTE1)/2
      ALFO(KTE2)=ALFO(K2)
      K=K2-1
      KK=KTE2-1
  210 ALFO(KK)=.5*(ALFO(K)+ALFO(K+1))
      ALFO(KK-1)=ALFO(K)
      KK=KK-2
      K=K-1
      IF (K.GE.KTE1) GO TO 210
      RETURN
      END



      SUBROUTINE SPLIF (M,N,S,F,FP,FPP,FPPP,KM,VM,KN,VN,MODE,FQM,IND)
C     CUBIC SPLINE FIT WITH PRESCRIBED END CONDITIONS
C     INTEGRAL PLACED IN FPPP IF MODE GREATER THAN 0
C     IND SET TO ZERO IF DATA ILLEGAL
      DIMENSION S(1), F(1), FP(1), FPP(1), FPPP(1)
      IND=0
      K=IABS(N-M)
      IF (K-1) 180,180,10
   10 K=(N-M)/K
      I=M
      J=M+K
      DS=S(J)-S(I)
      D=DS
      IF (DS) 20,180,20
   20 DF=(F(J)-F(I))/DS
      IF (KM-2) 30,40,50
   30 U=.5
      V=3.*(DF-VM)/DS
      GO TO 80
   40 U=0.
      V=VM
      GO TO 80
   50 U=-1.
      V=-DS*VM
      GO TO 80
   60 I=J
      J=J+K
```

```
      DS=S(J)-S(I)
      IF (D*DS) 180,180,70
   70 DF=(F(J)-F(I))/DS
      B=1./(DS+DS+U)
      U=B*DS
      V=B*(6.*DF-V)
   80 FP(I)=U
      FPP(I)=V
      U=(2.-U)*DS
      V=6.*DF+DS*V
      IF (J-N) 60,90,60
   90 IF (KN-2) 100,110,120
  100 V=(6.*VN-V)/U
      GO TO 130
  110 V=VN
      GO TO 130
  120 V=(DS*VN+FPP(I))/(1.+FP(I))
  130 B=V
      D=DS
  140 DS=S(J)-S(I)
      U=FPP(I)-FP(I)*V
      FPPP(I)=(V-U)/DS
      FPP(I)=U
      FP(I)=(F(J)-F(I))/DS-DS*(V+U+U)/6.
      V=U
      J=I
      I=I-K
      IF (J-M) 140,150,140
  150 I=N-K
      FPPP(N)=FPPP(I)
      FPP(N)=B
      FP(N)=DF+D*(FPP(I)+B+B)/6.
      IND=1
      IF (MODE) 180,180,160
  160 FPPP(J)=FQM
      V=FPP(J)
  170 I=J
      J=J+K
      DS=S(J)-S(I)
      U=FPP(J)
      FPPP(J)=FPPP(I)+.5*DS*(F(I)+F(J)-DS*DS*(U+V)/12.)
      V=U
      IF (J-N) 170,180,170
  180 RETURN
      END
```

```
      SUBROUTINE INTPL (MI,NI,SI,FI,M,N,S,F,FP,FPP,FPPP,MODE)
C     INTERPOLATION OF CUBIC SPLINE BY TAYLOR SERIES
C     ADDS CORRECTION FOR PIECEWISE CONSTANT FOURTH DERIVIATIVE
C     IF MODE GREATER THAN 0
      DIMENSION SI(1), FI(1), S(1), F(1), FP(1), FPP(1), FPPP(1)
      K=IABS(N-M)
      K=(N-M)/K
      I=M
      MIN=MI
      NIN=NI
      D=S(N)-S(M)
      IF (D*(SI(NI)-SI(MI))) 10,20,20
   10 MIN=NI
      NIN=MI
   20 KI=IABS(NIN-MIN)
      IF (KI) 40,40,30
   30 KI=(NIN-MIN)/KI
   40 II=MIN-KI
      C=0.
      IF (MODE) 60,60,50
   50 C=1.
   60 II=II+KI
      SS=SI(II)
   70 I=I+K
      IF (I-N) 80,90,80
   80 IF (D*(S(I)-SS)) 70,70,90
   90 J=I
      I=I-K
      SS=SS-S(I)
      FPPPP=C*(FPPP(J)-FPPP(I))/(S(J)-S(I))
      FF=FPPP(I)+.25*SS*FPPPP
      FF=FPP(I)+SS*FF/3.
      FF=FP(I)+.5*SS*FF
      FI(II)=F(I)+SS*FF
      IF (II-NIN) 60,100,60
  100 RETURN
      END
```

```
      SUBROUTINE THREED (IPLOT,SV,SM,CP,X,Y,TITLE,YA,AL,VLD,CL,CD,CHORDO
     1,XSCAL,PSCAL,LABEL,NIT,UC,VC,WC,NF1)
C     GENERATES THREE DIMENSIONAL PLOTS
C     GENERATES CALCOMP PLOTS ON CDC 6600
      COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
     1,ITE2(15),A0(129),A1(129),A2(129),A3(129),B0(12),B1(12),B2(12),B3(
     212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
```

```
     3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
     4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
     515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
     6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,F00,F01,F10,F11,
     7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
      DIMENSION X(1), Y(1), SV(1), SM(1), CP(1), TITLE(10), R(20), UC(1)
     1, VC(1), WC(1)
      M=1
      LX=NX/2+1
      MX=NX+1
      MY=NY+2
      IF (XSCAL.NE.0.) SCALX=.5*ABS(XSCAL)/CHORDO
      IF (PSCAL.GE.0.) SCALX=5./(Z(KTE2)-Z(KTE1))
      SCALP=-1.00
      IF (PSCAL.NE.0.) SCALP=-.5/ABS(PSCAL)
      TX=3.0
      SX=-SCALX*XC(KTE1)
      IF (IPLOT.NE.1) GO TO 10
      CALL PLOTSBL (10000,22HJEFF MCFADDEN  3210WWH)
   10 IPLOT=0
      CALL FRAME
      CALL PLOT (1.25,1.,-3)
      ASRAT=2.*(Z(KTE2)-Z(KTE1))**2/AWING
      ENCODE (60,90,R) FMACH,CL,VOLDRG,ASRAT
      CALL SYMBOL (.50,0.75,.14,R,0.,60)
      ENCODE (60,100,R)
      CALL SYMBOL (.50,1.25,.14,R,0.,60)
   20 CONTINUE
      K=1
      IF (KTE2.LT.10) K=2
   30 K=K+2
      IF (KTE2.LT.10) K=K-1
      IF (K.GT.KTE2) GO TO 70
      IF (K.LT.KTE1) GO TO 30
      I1=ITE1(K)
      I2=ITE2(K)
      CALL VELO (K,K,SV,SM,CP,X,Y,UC,VC,WC)
      SY=5.*(Z(K)-Z(KTE1))/(Z(KTE2)-Z(KTE1))+2.45
      SCP=5.*(Z(K)-Z(KTE1))/(Z(KTE2)-Z(KTE1))+2.75
      DO 40 I=I1,I2
      X(I)=SCALX*X(I)+SX
      Y(I)=SCALX*Y(I)+SY
   40 CP(I)=SCALP*CP(I)+SCP
      IF (M.EQ.2) GO TO 50
      N=I2-LX+1
      CALL LINE (X(LX),CP(LX),N,1,0,1,0.,1.,0.,1.)
      GO TO 30
   50 N=I2-I1+1
      DO 60 I=I1,I2
   60 X(I)=X(I)+TX
      N=I2-I1+1
      CALL LINE (X(I1),Y(I1),N,1,0,1,0.,1.,0.,1.)
```

```
      GO TO 30
   70 CONTINUE
      M=M+1
      IF (M.GT.2) GO TO 80
      GO TO 20
   80 CALL DRAGC (1,SCALX)
      IO=1
      CALL PLOT (-1.25,-1.,-3)
      RETURN
C
   90 FORMAT (4HM = ,F3.2,1H,2X,5HCL = ,F3.2,1H,2X,6HCDW = ,F5.4,1H,2X,4
     1HA = ,F3.1)
  100 FORMAT (22HUPPER SURFACE PRESSURE,5X,15HWING AND SHOCKS)
      END



      SUBROUTINE READQS (NQSTA,ZQSTA,PCQ1,PCQ2,PCQ3,Q1,Q2,Q3,Q4,PCS1,PCS
     12,DSURF,FMACH)
C     READS IN ASSIGNED SPEED DISTRIBUTION
      DIMENSION ZQSTA(1), PCQ1(1), PCQ2(1), PCQ3(1), Q1(1), Q2(1), Q3(1)
     1, Q4(1), PCS1(1), PCS2(1), DSURF(1)
      IREAD=9
      IWRIT=6
      AAO=1./FMACH**2+.2
      WRITE (IWRIT,20)
      READ (IREAD,40)
      READ (IREAD,40)
      READ (IREAD,50) FQSTA
      NQSTA=FQSTA
      DO 10 K=1,NQSTA
      READ (IREAD,40)
      READ (IREAD,50) ZQSTA(K),PCQ1(K),PCQ2(K),PCQ3(K),Q1(K),Q2(K),Q3(K)
     1,Q4(K)
      READ (IREAD,40)
      READ (IREAD,50) PCS1(K),PCS2(K),DSURF(K)
      WRITE (IWRIT,30) K,ZQSTA(K),PCQ1(K),PCQ2(K),PCQ3(K),Q1(K),Q2(K),Q3
     1(K),Q4(K),PCS1(K),PCS2(K),DSURF(K)
      Q1(K)=SQRT((AAO*Q1(K)**2)/(1.+.2*Q1(K)**2))
      Q2(K)=SQRT((AAO*Q2(K)**2)/(1.+.2*Q2(K)**2))
      Q3(K)=SQRT((AAO*Q3(K)**2)/(1.+.2*Q3(K)**2))
      Q4(K)=SQRT((AAO*Q4(K)**2)/(1.+.2*Q4(K)**2))
   10 CONTINUE
      RETURN
C
   20 FORMAT (55H1 PARAMETERS TO DEFINE THE ASSIGNED DESIGN MACH NUMBER
     113HDISTRIBUTION.,//,3H   K,9X,1HZ,6X,4HPCM1,6X,4HPCM2,6X,4HPCM3,8X,
```

```
   22HM1,8X,2HM2,8X,2HM3,8X,2HM4,6X,4HPCX1,6X,4HPCX2,5X,5HDSURF,/)
 30 FORMAT (1X,I2,11F10.5)
 40 FORMAT (1X)
 50 FORMAT (8F10.5)
    END



    SUBROUTINE SETQS (NE,NX,QPRE,SO,SOPRE,ITE1,ITE2,KTE1,KTE2,Z,ZQSTA,
   1AO,PCQ1,PCQ2,PCQ3,SI,FI,Q1,Q2,Q3,Q4,PCS1,PCS2,DSURF,NQSTA)
C   DEFINES ASSIGNED SPEED DISTRIBUTION BY EXPONENTIAL SPLINE
C   ALLOWS EASY CONSTRUCTION OF SHOCKLESS DISTRIBUTION USING E AND G
    DIMENSION QPRE(NE,1), SO(NE,1), SOPRE(NE,1), ITE1(1), ITE2(1), Z(1
   1), ZQSTA(1), AO(1), PCQ1(1), PCQ2(1), PCQ3(1), SI(1), FI(1), Q1(1)
   2, Q2(1), Q3(1), Q4(1), PCS1(1), PCS2(1), DSURF(1)
    DIMENSION X(4), Y(4), E(4), G(4), A(4), B(4), C(4), D(4)
    DATA ISET/0/
    BUMP(X)=16.*(X*(1.-X))**2
    E(1)=.007
    E(2)=.007
    E(3)=.007
    G(1)=0.
    G(2)=-3.
    G(3)=45.
    LX=NX/2+1
    MX=NX+1
    KM=2
    VM=0.
    KN=3
    VN=0.
    K2=2
    K=KTE1-1
 10 K=K+1
    I1=ITE1(K)
    I2=ITE2(K)
    ZO=Z(K)
    K2=K2-1
 20 K2=K2+1
    K1=K2-1
    Z1=ZQSTA(K1)
    Z2=ZQSTA(K2)
    IF (ZO.GT.Z2.AND.K2.LT.NQSTA) GO TO 20
    R1=(Z2-ZO)/(Z2-Z1)
    R2=1.-R1
    DLEN=AO(I2)-AO(LX)
    PX1=R1*PCQ1(K1)+R2*PCQ1(K2)
    PX2=R1*PCQ2(K1)+R2*PCQ2(K2)
```

```
      PX3=R1*PCQ3(K1)+R2*PCQ3(K2)
      X(1)=AO(LX)+PX1*DLEN
      X(2)=AO(LX)+PX2*DLEN
      X(3)=AO(LX)+PX3*DLEN
      X(4)=AO(I2)
      Y(1)=R1*Q1(K1)+R2*Q1(K2)
      Y(2)=R1*Q2(K1)+R2*Q2(K2)
      Y(3)=R1*Q3(K1)+R2*Q3(K2)
      Y(4)=R1*Q4(K1)+R2*Q4(K2)
      CALL SPTEN (4,X,Y,E,G,A,B,C,D,KM,VM,KN,VN)
      DO 30 I=1,MX
      QPRE(I,K)=0.
      IF (ISET.EQ.0) SOPRE(I,K)=SO(I,K)
   30 CONTINUE
      LS=I1
   40 LS=LS+1
      IF (AO(LS).LT.X(1)) GO TO 40
      NN=MX-LS+1
      DO 50 I=1,NN
      J=LS+I-1
   50 SI(I)=AO(J)
      CALL INTEN (NN,SI,FI,4,X,A,B,C,D,E,G)
      DO 60 I=LS,MX
      J=I-LS+1
   60 QPRE(I,K)=FI(J)
      DENO=1./FLOAT(LS-I1)
      DO 70 I=I1,LS
   70 QPRE(I,K)=FLOAT(I-I1)*DENO*Y(1)
      IF (ISET.EQ.1) GO TO 90
      PX2=R1*PCS2(K1)+R2*PCS2(K2)
      PX1=R1*PCS1(K1)+R2*PCS1(K2)
      DSOPRE=R1*DSURF(K1)+R2*DSURF(K2)
      X1=AO(LX)+PX1*DLEN
      X2=AO(LX)+PX2*DLEN
      I=I1
   80 I=I+1
      IF (AO(I).GT.X2) GO TO 90
      IF (AO(I).LT.X1) GO TO 80
      XX=(AO(I)-X1)/(X2-X1)
      SOPRE(I,K)=SO(I,K)-DSOPRE*BUMP(XX)
      GO TO 80
   90 IF (K.LT.KTE2) GO TO 10
      ISET=1
      RETURN
      END
```

```
      SUBROUTINE SPTEN (N,S,F,E,G,A,B,C,D,KM,VM,KN,VN)
C     COMPUTES EXPONENTIAL SPLINE WITH WEIGHTING FACTORS
C     E IS TENSION PARAMETER, SMALLER E PRODUCES LESS OSCILLATION
C     G IS WEIGHT FACTOR, LARGER G PRODUCES MORE SAG
      DIMENSION S(1), F(1), E(1), G(1), A(1), B(1), C(1), D(1)
      NM=N-1
      H=S(2)-S(1)
      HI=1./H
      SI=1./SINH(H/E(1))
      TI=SI*COSH(H/E(1))
      IF (KM-2) 10,20,30
   10 B(1)=E(1)*(E(1)*HI-TI)
      C(1)=E(1)*(SI-E(1)*HI)
      D(1)=VM+HI*(F(1)-F(2))+E(1)*G(1)*(SI-TI)+.5*H*G(1)
      GO TO 40
   20 B(1)=1.
      C(1)=0.
      D(1)=VM
      GO TO 40
   30 B(1)=-TI
      C(1)=SI
      D(1)=G(1)*(SI-TI)+VM*E(1)
   40 XX=1./B(1)
      D(1)=XX*D(1)
      DO 50 I=2,NM
      HM=H
      HIM=HI
      SIM=SI
      TIM=TI
      H=S(I+1)-S(I)
      HI=1./H
      SI=1./SINH(H/E(I))
      TI=SI*COSH(H/E(I))
      A(I)=E(I-1)*(SIM-HIM*E(I-1))
      B(I)=E(I)*(HI*E(I)-TI)+E(I-1)*(HIM*E(I-1)-TIM)
      C(I)=E(I)*(SI-HI*E(I))
      D(I)=HIM*(F(I)-F(I-1))+E(I-1)*G(I-1)*(SIM-TIM)+.5*HM*G(I-1)+HI*(F(
     1I)-F(I+1))+E(I)*G(I)*(SI-TI)+.5*H*G(I)
      C(I-1)=XX*C(I-1)
      XX=1./(B(I)-A(I)*C(I-1))
   50 D(I)=(D(I)-A(I)*D(I-1))*XX
      IF (KN-2) 60,70,80
   60 A(N)=E(NM)*(HI*E(NM)-SI)
      B(N)=E(NM)*(TI-HI*E(NM))
      D(N)=VN+HI*(F(NM)-F(N))-.5*H*G(NM)+E(NM)*G(NM)*(TI-SI)
      GO TO 90
   70 A(N)=0.
      B(N)=1.
      D(N)=VN
      GO TO 90
   80 A(N)=-SI/E(NM)
      B(N)=TI/E(NM)
```

```
      D(N)=VN+G(NM)*(TI-SI)/E(NM)
   90 C(NM)=XX*C(NM)
      XX=1./(B(N)-A(N)*C(NM))
      D(N)=(D(N)-A(N)*D(NM))*XX
      DO 100 J=2,N
      I=N+1-J
  100 D(I)=D(I)-C(I)*D(I+1)
      DI=D(1)
      DO 110 I=1,NM
      DIP=D(I+1)
      H=S(I+1)-S(I)
      SI=1./SINH(H/E(I))
      A(I)=F(I+1)-F(I)+(DI-DIP)*E(I)**2-.5*G(I)*H**2
      B(I)=F(I)+(G(I)-DI)*E(I)**2
      C(I)=SI*(DIP-G(I))*E(I)**2
      D(I)=SI*(DI-G(I))*E(I)**2
  110 DI=DIP
      RETURN
      END



      SUBROUTINE INTEN (NX,SI,FI,N,S,A,B,C,D,E,G)
C     COMPUTES INTERPOLATED VALUES FROM EXPONENTIAL SPLINE
      DIMENSION SI(1), FI(1), S(1), A(1), B(1), C(1), D(1), E(1), G(1)
      I=0
      J=1
      JP=J+1
   10 I=I+1
   20 IF (SI(I).LT.S(JP)) GO TO 30
      IF (JP.EQ.N) GO TO 30
      J=J+1
      JP=J+1
      GO TO 20
   30 HI=1./(S(JP)-S(J))
      EI=1./E(J)
      FI(I)=B(J)+HI*A(J)*(SI(I)-S(J))+C(J)*SINH(EI*(SI(I)-S(J)))+D(J)*SI
     1NH(EI*(S(JP)-SI(I)))+.5*G(J)*(SI(I)-S(J))**2
      IF (I.LT.NX) GO TO 10
      RETURN
      END
```

```
C     THIS VERSION OF SUBROUTINE YSWEEP IS VECTORIZED AND CAN BE INVOKED
C     BY SETTING FSWEEP TO -1.0 ON THE APPROPRIATE INPUT CARD
      SUBROUTINE VYSWEEP
C     ROW RELAXATION
      COMMON G(129,12,15),SO(129,15),EO(131),ZO(131),IV(129,15),ITE1(15)
     1,ITE2(15),AO(129),A1(129),A2(129),A3(129),BO(12),B1(12),B2(12),B3(
     212),Z(15),C1(15),C2(15),C3(15),XC(15),XZ(15),XZZ(15),YC(15),YZ(15)
     3,YZZ(15),NX,NY,NZ,KTE1,KTE2,ISYM,KSYM,SCAL,SCALZ,YAW,CYAW,SYAW,ALP
     4HA,CA,SA,FMACH,N1,N2,N3,IO,NDES,TSTEP,EPS1,QPRE(129,15),SOPRE(129,
     515),NQSTA,ZQSTA(15),PCQ1(15),PCQ2(15),PCQ3(15),QQ1(15),QQ2(15),QQ3
     6(15),QQ4(15),PCS1(15),PCS2(15),DSURF(15),RDQ,RDSO,FOO,FO1,F10,F11,
     7NDQ,IQ,KQ,AWING,VOLDRG,IDRGPLT(129,15),SECDRG(15)
      COMMON /FLO/ STRIP,P1,P2,P3,BETA,FR,IR,JR,KR,DG,IG,JG,KG,NS
      COMMON /SWP/ DXYZ(129),GK1(129,15),GK2(129,15),SX(129),SZ(129),SXX
     1(129),SXZ(129),SZZ(129),RO(129),R1(129),C(129),D(129),G10(15),G20(
     215),G30(15),G40(15),G1(15),G2(15),I1,I2,K,L,NO,LX,MX,KY,MY,T1,AAO,
     3Q1,Q2,TYAW,S1
      COMMON /DIM/ NX1,NY1,NZ1,FDIM
      COMMON /CRAY/ KA
      COMMON /VECT/ YP(129),SAVE(129),TEMP2(129),TEMP(129),TEMP1(129),AJ
     1(129),H(129),FH(129),AZ(129),BZ(129),CZ(129),DZ(129),DGI(129),DGJ(
     2129),DGK(129),U(129),V(129),W(129),AU(129),AV(129),QQ(129),HOLD1(1
     329),HOLD2(129),HOLD3(129),HOLD4(129),HOLD5(129),HZ(129),AA(129),FX
     4X(129),FYY(129),FXY(129),BV(129),UU(129),VV(129),WW(129),UV(129),U
     5W(129),VW(129),AZZ(129),AXX(129),AXZ(129),R(129),AXT(129),AYT(129)
     6,AZT(129),DGII(129),DGJJ(129),DGIJ(129),DGIK(129),DGJK(129),AC(129
     7),AB(129),AYY(129),AYZ(129),BP(129),BM(129),B(129),AXY(129),CG(129
     8),DGKK(129),A(129),S(129)
      BETX=.01
      BETY=.15
      BETZ=.1
      BSCAL=1./(1.+FDIM)
      BSCAL1=1./(2.*(1.+FDIM))
      J1=2
      IF (FMACH.GE.1.) J1=3
      C(I1-1)=0.
      D(I1-1)=0.
      DO 10 I=I1,I2
      RO(I)=1.
      R1(I)=1.
      GK1(I,1)=G(I,1,L)
   10 GK1(I,J1-1)=G(I,J1-1,L)
      J=J1
      I3=I2
   20 BC=-T1*B1(J)*C1(K)
      DO 30 I=I1,I3
      YP(I)=SO(I,K)+BO(J)
```

```
      SAVE(I)=1.0-RO(I)
      TEMP2(I)=YP(I)*YP(I)
      TEMP(I)=AO(I)*AO(I)
      AJ(I)=SAVE(I)+TEMP2(I)+TEMP(I)
   30 CONTINUE
      DO 40 I=I1,I3
      H(I)=RO(I)/AJ(I)
      FH(I)=RO(I)*AJ(I)
      TEMP1(I)=AO(I)*(4.*TEMP2(I)-FH(I))
      TEMP2(I)=YP(I)*(4.*TEMP(I)-FH(I))
      AT=XZ(K)*XZ(K)-YZ(K)*YZ(K)
      BT=(XZ(K)+XZ(K))*YZ(K)
      AZ(I)=-AO(I)*XZ(K)-YP(I)*YZ(K)
      BZ(I)=-AO(I)*YZ(K)+YP(I)*XZ(K)
      TEMP(I)=H(I)*H(I)
      CZ(I)=TEMP(I)*(TEMP1(I)*AT-TEMP2(I)*BT)-AO(I)*XZZ(K)-YP(I)*YZZ(K)
      DZ(I)=TEMP(I)*(TEMP2(I)*AT+TEMP1(I)*BT)-AO(I)*YZZ(K)+YP(I)*XZZ(K)
   40 CONTINUE
      DO 50 I=I1,I3
      DGI(I)=G(I+1,J,L)-G(I-1,J,L)
      DGJ(I)=G(I,J+1,L)-GK1(I,J-1)
      DGK(I)=G(I,J,L+1)-GK1(I,J)
   50 CONTINUE
      DO 60 I=I1,I3
      TEMP1(I)=A1(I)*DGI(I)
      TEMP2(I)=-B1(J)*DGJ(I)
      U(I)=TEMP1(I)-SX(I)*TEMP2(I)+CA*AO(I)+SA*YP(I)
      V(I)=TEMP2(I)+SA*AO(I)-CA*YP(I)
      W(I)=RO(I)*(C1(K)*DGK(I)-SZ(I)*TEMP2(I)+SYAW+CA*XZ(K)+SA*YZ(K)+H(I
     1)*(U(I)*AZ(I)+V(I)*BZ(I)))
      AU(I)=U(I)+W(I)*AZ(I)
      AV(I)=V(I)+W(I)*BZ(I)
      TEMP(I)=H(I)*(U(I)*U(I)+V(I)*V(I))
      QQ(I)=TEMP(I)+W(I)*W(I)
   60 CONTINUE
      DO 70 I=I1,I3
      HOLD1(I)=.2*QQ(I)
      AA(I)=DIM(AAO,HOLD1(I))
      HZ(I)=AZ(I)*SX(I)-BZ(I)+FH(I)*SZ(I)
      FXX(I)=1.0+H(I)*AZ(I)*AZ(I)
      FYY(I)=1.0+SX(I)*SX(I)+H(I)*HZ(I)*HZ(I)
      FXY(I)=SX(I)+H(I)*AZ(I)*HZ(I)
      BV(I)=AV(I)-AU(I)*SX(I)-FH(I)*W(I)*SZ(I)
   70 CONTINUE
      DO 80 I=I1,I3
      UU(I)=H(I)*AU(I)*AU(I)
      VV(I)=H(I)*BV(I)*BV(I)
      WW(I)=FH(I)*W(I)*W(I)
      UV(I)=H(I)*AU(I)*BV(I)
      UW(I)=AU(I)*W(I)
      VW(I)=BV(I)*W(I)
      AXX(I)=R1(I)*(FXX(I)*AA(I)-UU(I))
```

```
      AZZ(I)=FH(I)*AA(I)-WW(I)
      AXZ(I)=(2.0*RO(I)*(AZ(I)*AA(I)-UW(I)))
   80 CONTINUE
      DO 90 I=I1,I2
      HOLD1(I)=-TEMP2(I)*(AXX(I)*SXX(I)+AZZ(I)*SZZ(I)+AXZ(I)*SXZ(I))
      HOLD2(I)=(AA(I)*(CZ(I)*TEMP1(I)+(DZ(I)-SX(I)*CZ(I))*TEMP2(I)))*RO(
     1I)
      HOLD3(I)=CA*(AU(I)*AU(I)-AV(I)*AV(I))+(SA+SA)*AU(I)*AV(I)
      HOLD4(I)=TEMP(I)*(U(I)*AO(I)+V(I)*YP(I)+2.0*W(I)*(AO(I)*AZ(I)+YP(I
     1)*BZ(I)))
      HOLD5(I)=-WW(I)*(CA*XZZ(K)+SA*YZZ(K))-W(I)*W(I)*(U(I)*CZ(I)+V(I)*D
     1Z(I))
      R(I)=HOLD1(I)+T1*(HOLD2(I)-H(I)*(HOLD3(I)-HOLD4(I))+HOLD5(I))
   90 CONTINUE
      DO 100 I=I1,I3
      AXT(I)=AU(I)*A1(I)
      AXT(I)=ABS(AXT(I))
      AYT(I)=BV(I)*B1(J)
      AYT(I)=ABS(AYT(I))
      AZT(I)=FH(I)*W(I)*C1(K)
      AZT(I)=ABS(AZT(I))
      SAVE(I)=AMAX1(AXT(I),AYT(I),AZT(I),(1.-RO(I)))
      HOLD1(I)=RO(I)*BETA*AA(I)/SAVE(I)
      AXT(I)=AXT(I)*HOLD1(I)
      AYT(I)=AYT(I)*HOLD1(I)
      AZT(I)=AZT(I)*HOLD1(I)
  100 CONTINUE
      DO 110 I=I1,I3
      DGII(I)=G(I+1,J,L)-G(I,J,L)-G(I,J,L)+G(I-1,J,L)+A3(I)*DGI(I)
      DGJJ(I)=G(I,J+1,L)-G(I,J,L)-G(I,J,L)+G(I,J-1,L)-B3(J)*DGJ(I)
      DGKK(I)=G(I,J,L+1)-G(I,J,L)-G(I,J,L)+G(I,J,L-1)+C3(K)*DGK(I)
      DGIJ(I)=G(I+1,J+1,L)-G(I-1,J+1,L)-G(I+1,J-1,L)+G(I-1,J-1,L)
      DGIK(I)=G(I+1,J,L+1)-G(I+1,J,L-1)-G(I-1,J,L+1)+G(I-1,J,L-1)
      DGJK(I)=G(I,J+1,L+1)-G(I,J-1,L+1)-G(I,J+1,L-1)+G(I,J-1,L-1)
      AC(I)=T1*A1(I)*C1(K)
      AB(I)=-T1*A1(I)*B1(J)
      AXX(I)=AXX(I)*A2(I)
      AYY(I)=(FYY(I)*AA(I)-VV(I))*B2(J)
      AZZ(I)=AZZ(I)*C2(K)
      AXY(I)=-R1(I)*(FXY(I)*AA(I)+UV(I))*(AB(I)+AB(I))
      AXZ(I)=AXZ(I)*AC(I)
      AYZ(I)=-RO(I)*(HZ(I)*AA(I)+VW(I))*(BC+BC)
      BP(I)=AXX(I)
      BM(I)=AXX(I)
      B(I)=-AXX(I)-AXX(I)-Q1*(AYY(I)+AZZ(I))
      SAVE(I)=AXX(I)*DGII(I)+AYY(I)*DGJJ(I)+AZZ(I)*DGKK(I)+AXY(I)*DGIJ(I
     1)+AYZ(I)*DGJK(I)+AXZ(I)*DGIK(I)
  110 CONTINUE
      DO 120 I=I1,I3
      IF (QQ(I).LT.AA(I)) GO TO 120
      NS=NS+1
      S(I)=SIGN(1.,U(I))
```

```
      IM=I-IFIX(S(I))
      IMM=IM-IFIX(S(I))
      AXX(I)=UU(I)*A2(I)
      AYY(I)=VV(I)*B2(J)
      AZZ(I)=WW(I)*C2(K)
      AXY(I)=8.*S(I)*UV(I)*AB(I)
      AXZ(I)=8.*S(I)*UW(I)*AC(I)
      AYZ(I)=8.*VW(I)*BC
      HOLD1(I)=(FXX(I)*QQ(I)-UU(I))*A2(I)
      HOLD2(I)=(FYY(I)*QQ(I)-VV(I))*B2(J)
      HOLD3(I)=(FH(I)*QQ(I)-WW(I))*C2(K)
      HOLD4(I)=-(FXY(I)*QQ(I)+UV(I))*(AB(I)+AB(I))
      HOLD5(I)=(AZ(I)*QQ(I)-UW(I))*(AC(I)+AC(I))
      TEMP(I)=-(HZ(I)*QQ(I)+VW(I))*(BC+BC)
      TEMP1(I)=AA(I)/QQ(I)
      TEMP2(I)=HOLD1(I)*DGII(I)+HOLD2(I)*DGJJ(I)+HOLD3(I)*DGKK(I)+HOLD4(
     1I)*DGIJ(I)+TEMP(I)*DGJK(I)+HOLD5(I)*DGIK(I)
      DGII(I)=G(I,J,L)-G(IM,J,L)-G(IM,J,L)+G(IMM,J,L)+A3(I)*DGI(I)
      DGJJ(I)=G(I,J,L)-G(I,J-1,L)-G(I,J-1,L)+GK1(I,J-2)-B3(J)*DGJ(I)
      DGKK(I)=G(I,J,L)-G(I,J,L-1)-G(I,J,L-1)+GK2(I,J)+C3(K)*DGK(I)
      DGIJ(I)=G(I,J,L)-G(IM,J,L)-G(I,J-1,L)+G(IM,J-1,L)
      DGIK(I)=G(I,J,L)-G(I,J,L-1)-G(IM,J,L)+G(IM,J,L-1)
      DGJK(I)=G(I,J,L)-G(I,J,L-1)-G(I,J-1,L)+G(I,J-1,L-1)
      TEMP(I)=AXX(I)*DGII(I)+AYY(I)*DGJJ(I)+AZZ(I)*DGKK(I)+AXY(I)*DGIJ(I
     1)+AYZ(I)*DGJK(I)+AXZ(I)*DGIK(I)
      B(I)=.5*(TEMP1(I)-1.)*(AXX(I)+AXX(I)+AXZ(I)+AXY(I))
      BP(I)=TEMP1(I)*HOLD1(I)-(1.-S(I))*B(I)
      BM(I)=TEMP1(I)*HOLD1(I)-(1.0+S(I))*B(I)
      B(I)=-TEMP1(I)*(HOLD1(I)+HOLD1(I)+Q2*(HOLD2(I)+HOLD3(I)))+(TEMP1(I
     1)-1.)*(2.*(AXX(I)+AYY(I)+AZZ(I))+AXY(I)+AYZ(I)+AXZ(I))
      SAVE(I)=(TEMP1(I)-1.)*TEMP(I)+TEMP1(I)*TEMP2(I)
  120 CONTINUE
      DO 130 I=I1,I3
  130 R(I)=R(I)+SAVE(I)
      DO 140 I=I1,I3
      IF (ABS(R(I)).LE.ABS(FR)) GO TO 140
      FR=R(I)
      IR=I
      JR=J
      KR=K
  140 CONTINUE
      DO 150 I=I1,I3
      R(I)=R(I)-AYT(I)*(GK1(I,J-1)-G(I,J-1,L))-AZT(I)*(GK1(I,J)-G(I,J,L-
     11))
      B(I)=B(I)-AXT(I)-AYT(I)-AZT(I)
      BM(I)=BM(I)+AXT(I)
  150 CONTINUE
      DO 160 I=I1,I3
      B(I)=1.0/(B(I)-BM(I)*C(I-1))
      C(I)=B(I)*BP(I)
  160 D(I)=B(I)*(R(I)-BM(I)*D(I-1))
      I=I3
```

```
      CG(I3+1)=0.
      DO 170 M=I1,I3
      CG(I)=D(I)-C(I)*CG(I+1)
      GK2(I,J)=GK1(I,J)
      GK1(I,J)=G(I,J,L)
      G(I,J,L)=G(I,J,L)-CG(I)
  170 I=I-1
      I=I3
      DO 180 M=I1,I3
      IF (ABS(CG(I)).LE.ABS(DG)) GO TO 180
      DG=CG(I)
      IG=I
      JG=J
      KG=K
  180 I=I-1
      J=J+1
      IF (J-KY) 20,190,210
  190 IF (I2.GT.ITE2(K)) I3=ITE2(K)
      IF (ITE2(K).EQ.MX) I3=LX
      DO 200 I=I1,I3
      LV=IABS(1-IABS(IV(I,K)))
      RO(I)=AMINO(LV,IABS(IV(I,K)))
  200 R1(I)=LV
      GO TO 20
  210 N=NO
      I=LX+1
      IF (K.LT.KTE1.OR.K.GT.KTE2) GO TO 230
      IO=NX+2-I3
      DO 220 I=IO,I3
      AJ(I)=1.-RO(I)+AO(I)*AO(I)+SO(I,K)*SO(I,K)
      H(I)=RO(I)/AJ(I)
      FH(I)=RO(I)*AJ(I)
      AZ(I)=-AO(I)*XZ(K)-SO(I,K)*YZ(K)
      BZ(I)=-AO(I)*YZ(K)+SO(I,K)*XZ(K)
      HZ(I)=AZ(I)*SX(I)-BZ(I)+FH(I)*SZ(I)
      FYY(I)=1.+SX(I)*SX(I)+H(I)*HZ(I)*HZ(I)
      FXY(I)=SX(I)+H(I)*AZ(I)*HZ(I)
      DGI(I)=G(I+1,KY,L)-G(I-1,KY,L)
      DGK(I)=G(I,KY,L+1)-GK2(I,KY)
      V(I)=SA*AO(I)-CA*SO(I,K)
      U(I)=A1(I)*DGI(I)+CA*AO(I)+SA*SO(I,K)
      W(I)=C1(K)*DGK(I)+SYAW+CA*XZ(K)+SA*YZ(K)
  220 G(I,KY+1,L)=G(I,KY-1,L)+(V(I)*(1.-H(I)*BZ(I)*HZ(I))-U(I)*FXY(I)-W(
     1I)*HZ(I))/(FYY(I)*B1(KY))
      I=IO
      IF (IO.NE.ITE1(K)) GO TO 230
      E=G(I3,KY,L)-G(IO,KY,L)
      NO=NO+1
      EO(NO)=EO(NO)+P3*(E-EO(NO))
      N=NO
  230 IF (I.LE.I1) GO TO 270
      I=I-1
```

```
      E=0.
      IF (IV(I,K).NE.1) GO TO 260
      ZZ=Z(K)-TYAW*(XC(K)+S1*AO(I)*AO(I))
  240 IF (ZZ.GE.ZO(N-1)) GO TO 250
      N=N-1
      GO TO 240
  250 RV=(ZZ-ZO(N-1))/(ZO(N)-ZO(N-1))
      E=RV*EO(N)+(1.-RV)*EO(N-1)
  260 M=NX+2-I
      G(I,KY+1,L)=G(M,KY-1,L)-E
      G(M,KY+1,L)=G(I,KY-1,L)+E
      GK2(M,KY)=GK1(M,KY)
      GK1(M,KY)=G(M,KY,L)
      G(M,KY,L)=G(I,KY,L)+E
      GO TO 230
  270 CONTINUE
      DO 280 I=2,NX
  280 G(I,J1-1,L)=(1.-BETY/BSCAL1)*G(I,J1,L)
      DO 290 J=1,MY
      G(I1-1,J,L)=(1.-BETX/BSCAL)*G(I1,J,L)
  290 G(I2+1,J,L)=(1.-BETX/BSCAL)*G(I2,J,L)
      RETURN
      END
```

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-3662		
4. Title and Subtitle THE NYU INVERSE SWEPT WING CODE		5. Report Date January 1983 6. Performing Organization Code
7. Author(s) F. Bauer, P. Garabedian, and G. McFadden		8. Performing Organization Report No. 10. Work Unit No.
9. Performing Organization Name and Address Courant Institute of Mathematical Sciences New York University New York, NY 10012		11. Contract or Grant No. NSG-1579
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546		13. Type of Report and Period Covered Contractor Report 14. Sponsoring Agency Code

15. Supplementary Notes

Langley Technical Monitor: Jerry C. South, Jr. Topical Report

This report was partially funded by NASA Ames Research Center under Grant NGR 33-016-201

16. Abstract

An inverse swept wing code is described that is based on the widely used transonic flow program FLO22. The new code incorporates a free boundary algorithm permitting the pressure distribution to be prescribed over a portion of the wing surface. A special routine is included to calculate the wave drag, which can be minimized in its dependence on the pressure distribution. An alternate formulation of the boundary condition at infinity has been introduced to enhance the speed and accuracy of the code. A FORTRAN listing of the code and a listing of a sample run are presented. There is also a user's manual as well as glossaries of input and output parameters.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Supercritical Wing Design Shock Waves Wave Drag	FEDD Distribution Subject Category 02

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	104	

Available: NASA's Industrial Applications Centers